Universidade de Aveiro
2016
Departamento de Eletrónica, Telecomunicações e Informática

Fábio André de Jesus Marques

Configuração Remota para Sistemas de Iluminação Pública Inteligente

Universidade de Aveiro
2016
Departamento de Eletrónica, Telecomunicações e Informática

Fábio André de Jesus Marques

Configuração Remota para Sistemas de Iluminação Pública Inteligente

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações realizada sob a orientação científica do Dr. Luís Filipe Mesquita Nero Moreira Alves, professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro e do Dr. Pedro Nicolau Faria da Fonseca, professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri

presidente

Prof. Doutor José Carlos Esteves Duarte Pedro
Professor Catedrático do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

Mestre Nuno Rafael Mendonça Lourenço
Especialista, Think Control, Unipessoal Lda

Prof. Doutor Luís Filipe Mesquita Nero Moreira Alves
Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

agradecimentos

Seria impossível mencionar todos aqueles que direta ou indiretamente foram importantes nesta etapa que agora termina. Não menosprezando a sua importância, vou dirigir os meus agradecimentos de forma mais direta aqueles que estiveram mais envolvidos nesta fase final do percurso académico.

Em primeiro lugar, aos meus orientadores, o Professor Doutor Luís Filipe Mesquita Nero Moreira Alves e o Professor Doutor Pedro Nicolau Faria da Fonseca pela oportunidade dada, orientação e tolerância ao longo do trabalho.
À Universidade de Aveiro, em particular ao DETI e ao IT, pelos meios disponibilizados e pela qualidade de ensino prestada.
Ao Professor Doutor João Paulo Barraca, pela disponibilidade e suporte dado nos assuntos relacionados com o envio de informação para a plataforma *web*.

A todos os meus amigos e colegas de curso, que ao longo dos anos partilharam comigo toda esta experiência única.

À empresa Engenho de Mestre, Lda, em particular ao Engenheiro Paulo Alves pela compreensão e disponibilidade demonstradas ao longo dos últimos meses, que permitiram conciliar o trabalho com a elaboração da presente dissertação.
Aos meus colegas de trabalho, em especial ao Engenheiro Bruno Martins e ao Engenheiro Pedro Santos pela ajuda e apoio prestados.

À minha família, em especial aos meus pais Isabel e Fernando, pela oportunidade dada, e pelo esforço que fizeram para me fornecerem a estabilidade necessária, mas também pelo carinho e compreensão demonstrados, e aos meus irmãos Samuel e Inês, pelo estímulo e companheirismo.

Por fim à Neide, pelo incentivo e persistência, mas também pelo apoio incondicional e amor demonstrados nos momentos mais difíceis.

A todos vocês...um Muito Obrigado!

palavras-chave

Iluminação pública inteligente, sensores, comunicação, gestão remota, LITES

resumo

A presente dissertação, aborda a problemática da eficiência energética nos sistemas de iluminação pública, responsáveis por uma grande fatia do consumo energético global. Inserida no projeto LITES, esta dissertação visa continuar o trabalho efetuado nos últimos anos em um dos pilotos do projeto, neste caso na ponte pedonal do Crasto, situada no Campus da Universidade de Aveiro. Em relação ao sistema atual, pretende-se debelar algumas das limitações que apresenta na sua configuração, desde os reduzidos parâmetros passíveis de serem alterados, à necessidade de deslocação ao local e dificuldade para a sua realização. Desta forma, numa fase inicial, o objetivo será criar uma luminária de teste, compatível com todo o sistema implementado e que permita além da monitorização, a configuração remota de alguns parâmetros do sistema. Para isso, será efetuado um estudo de diversos sensores de interesse, bem como protocolos de comunicação possíveis de serem utilizados neste sistema. A criação e implementação de um sensor de movimento híbrido constituído por um sensor passivo e um ativo, em substituição do sensor de movimento comercial atualmente utilizado, será um dos pontos de relevo desta dissertação. No final, será feita uma apresentação do sistema desenvolvido e uma discussão sobre os resultados obtidos. Finalmente, será feita uma comparação entre o sistema implementado e este novo sistema.

keywords

Intelligent Street Lighting, sensors, communication, remote management, LITES

abstract

The present dissertation deals with the problematic question about the energy efficiency in the public lighting systems, responsible for a large share of the global energy consumption. Inserted in the LITES project, this thesis aims to continue the work accomplished in this recent years in one of the project's pilot, the Crasto's pedestrian bridge located on the University Campus in Aveiro. Compared to the current system, it is intended to tackle some of the limitations found on its configuration, as the reduced parameters possible to change to the need of travel to the location and the difficulty to complete the task. This way, in an initial stage, the main goal is to create a compatible prototype with all the implemented system and allowing in addition to monitoring, the remote configuration of some system parameters. To achieve this, it will be made a study of various sensors of interest as well as possible communication protocols to be used in this system. The creation and implementation of a hybrid motion sensor, composed by a passive and active motion sensor, replacing the commercial one currently used, it will be one of the major points of this dissertation. At the end, it will be made a presentation of the developed system and a discussion of the results. Lastly, it will also be made a comparison between the new and the implemented system and conclusions will be taken.

Conteúdos

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

A

ADC – Analog-to-Digital Converter
ATNoG – Advanced Telecommunications and Network Group

B

BLE – Bluetooth Low Energy

C

CAN – Controller Area Network

D

E

EIA – Electronic Industries Alliance

F

FET – Field Effect Transistor
FoV – Field of View
FIFO – First In, First Out

G

H

I

I^2C –Inter-Integrated Circuit
IEEE – Institute of Electrical and Electronics Engineers
IP – Iluminação pública.
ISM – Industrial, Scientific and Medical
IDE – Integrated Development Environment

J

K

L

LDR – Light Dependent Resistor
LED – Light Emitting Diode
LITES – LED-based Intelligent Street Lighting for Energy Saving
LVDT – Linear Variable Differential Transformer
Laser – Light Amplification by Stimulated Emission of Radiation

M
N

NTC –Negative Temperature Coefficient

O
P

PIR – Passive Infrared
PLC – Power Line Communication
PTC – Positive Temperature Coefficient
PWM – Pulse-Width Modulation
PCB – Printed Circuit Board

Q
R

RTD – Resistance Temperature Detector
RISC – Reduced Instruction Set Computer

S

SPI – Serial Peripheral Interface
SSL – Solid State Lighting

T

TIA – Telecommunications Industry Association

U

UC – Unidade de controlo

V
W

WLAN – Wireless Local Area Network

X
Y
Z

Capítulo 1

1 - Introdução

A iluminação pública é uma das formas mais básicas de segurança durante o período noturno. Mais do que isso, transmite um certo conforto na ausência de luz natural, possibilitando a realização das mais diversas atividades nesse período. Além das vias públicas pedestres e rodoviárias, também locais de lazer como monumentos ou áreas de grande concentração de pessoas, são excelentes exemplos de locais onde a iluminação pública (IP) é indispensável nos dias de hoje. O aumento do uso de iluminação nas áreas e espaços referidos, fez aumentar gigantescamente o consumo energético (muito devido à baixa eficiência dos sistemas de IP utilizados) trazendo encargos financeiros muito elevados, que face à conjuntura económica que se verifica em muitos países como Portugal, tornaram-se praticamente insustentáveis para os municípios. Face a este cenário, muitos começaram a fazer cortes e a forma encontrada para poupar no consumo energético foi desligando a iluminação pública durante maiores períodos de tempo em zonas menos movimentadas, e desligando algumas luminárias, diminuindo a segurança e comodidade da população. É portanto imprescindível, que estes sistemas se tornem os mais eficientes possíveis, para que cenários como estes não sejam recorrentes.

Figura 1 - Iluminação pública com tecnologia LED [1].

1.1 - Enquadramento e motivação

O primeiro passo para a melhoria da eficiência energética nos sistemas de IP foi dado com a introdução de dispositivos de iluminação de estado sólido (SSL, do inglês *Solid State Lighting*), mais conhecidos como díodos emissores de luz ou LED's (*Light Emitting Diodes*), em substituição às lâmpadas convencionais utilizadas, como as incandescentes, fluorescentes ou de descarga, por exemplo. Esta tecnologia traz uma melhoria significativa na eficiência energética bem como uma importante redução na emissão de CO_2. Apesar dos benefícios, esta mudança tem sido lenta, muito por culpa do investimento necessário. No entanto, devido ao preço cada vez mais competitivo desta tecnologia, aliado às metas traçadas por algumas entidades, como a Estratégia Europa 2020 [2], que visam baixar quer a emissão de gases quer o consumo energético até 2020, é assim expectável que nos próximos anos se verifiquem maiores mudanças.

A implementação da tecnologia LED é uma grande evolução, e um grande passo para a melhoria da eficiência energética dos sistemas de IP. No entanto, cabe ao ser humano continuar a procurar e desenvolver novas formas de poupar energia e diminuir o impacto ambiental. Uma das formas de o fazer é dotando o sistema de inteligência, como unidades de controlo e sensores, permitindo assim uma gestão mais eficiente. Sendo um sistema inteligente e dotado de vários recursos, faz sentido que este permita a monitorização de toda a informação relevante recolhida, bem como a alteração e configuração de diversos parâmetros do sistema, desde sensoriais a perfis de iluminação. É neste ponto que o tema desta dissertação se insere, uma vez que num sistema como este, torna-se inviável que a pessoa responsável tenha que se deslocar ao local, alterando individualmente a configuração de cada luminária.

A presente dissertação está inserida no âmbito do projeto europeu LITES (*Led-based Intelligent Street Lighting for Energy Saving*) [3] que visa a criação de um sistema de iluminação pública recorrendo à tecnologia LED e com inteligência integrada, tentando assim potenciar ao máximo a eficiência energética. O projeto entretanto já terminado, teve como objetivo, implementar o sistema desenvolvido em locais piloto sendo um deles em Portugal, na cidade de Aveiro, mais concretamente na ponte pedonal do Crasto, situado no Campus da Universidade de Aveiro. Foram vários os motivos para a escolha deste local, para além de se situar no Campus da Universidade. É um local um pouco remoto e escuro mas de grande importância para os alunos. Está sujeito a elevados níveis de humidade, salinidade, nevoeiro e ventos relativamente fortes, que pelo facto de ser uma ponte, o sistema fica sujeito a vibrações que podem afetar o seu bom funcionamento. São estas condições adversas que tornam este local num bom piloto de testes para um sistema de IP.

Este projeto veio no seguimento do plano de ação adotado pela Comissão Europeia em 2006 [2], que serve de referência à política comunitária no que diz respeito ao consumo de energia. As metas traçadas, apontam à redução em 20% da emissão de gases de efeito de estufa em relação a 1990, alcançar uma melhoria de 20% da eficiência energética e conseguir que 20% da energia consumida pela União Europeia provenha de fontes de energia renováveis [4].

O projeto pretendia fornecer um serviço de iluminação pública inteligente, de acordo com a norma EN13201. Isto significa que a solução desenvolvida poderá ser

instalada em ruas secundárias, estradas de acesso, acessos a zonas comerciais, loteamentos, vias pedonais, ciclovias e zonas residenciais. De igual forma, apresentará conformidade com as normas EN60598-1 e EN60598-3 relativas aos regulamentos elétricos para luminárias, requisitos gerais e condições de teste [3].

Realizaram-se diversos trabalhos no âmbito deste projeto, no entanto aquele cujos resultados são mais visíveis, pode ser observado na ponte do Crasto, onde se encontra instalado e em funcionamento um sistema de IP inteligente. Este sistema permite não só uma poupança energética, uma vez que a iluminação é significativamente reduzida na ausência de movimento, como também permite dar algum conforto e segurança às pessoas que a utilizam, na medida em que existe sempre alguma iluminação mesmo quando não existe circulação.

Embora já existam no mercado diversos sistemas de IP com inteligência integrada, o seu custo é muito elevado. Há portanto, a necessidade de uma investigação e desenvolvimento continuados na procura de melhores resultados a preços mais baixos, permitindo assim uma mais rápida e fácil substituição dos antigos sistemas de IP, e consequentemente uma diminuição do impacto ambiental.

1.2 - Objetivos

A presente dissertação tem como objetivos propostos, o estudo e desenvolvimento de um módulo de controlo para uma luminária com as seguintes características:

- Recolha de dados através de diversos sensores;
- Transmissão da informação recolhida para um *gateway* que posteriormente enviará essa informação para a *web* onde poderá ser acedida através de uma plataforma já desenvolvida;
- Configuração remota;

Deste modo, as tarefas propostas para a sua elaboração são:

- Análise do sistema já desenvolvido e atualmente em uso no piloto da ponte pedonal do Crasto [6];
- Familiarização com os conceitos associados a sistemas de IP;
- Estudo das arquiteturas de controlo existentes em sistemas de IP inteligentes;
- Estudo e definição dos elementos sensoriais bem como dos protocolos de comunicação a integrar no sistema:

- Desenvolvimento de um sensor de movimento híbrido constituído por um sensor de infravermelhos passivo (PIR) e um sensor de micro-ondas;

- Otimizar e/ou desenvolver os módulos necessários para o sistema;

- Produção e implementação de um protótipo experimental;

- Testes e comparação com os módulos em uso;

1.3 - Estrutura

A presente dissertação encontra-se estruturada em 6 capítulos.

Ao longo do capítulo 2 é feito um levantamento do estado da arte da iluminação pública, contendo um pouco da história e evolução destes sistemas e também as diversas topologias utilizadas.

No capítulo 3 são identificadas as partes constituintes de um sistema de IP e realizado um estudo e apresentação dos diversos sensores e protocolos de comunicação que se podem utilizar.

O capítulo 4 apresenta o trabalho experimental realizado, desde o projeto e desenvolvimento do sensor híbrido, à concessão final da PCB a implementar na luminária piloto. No final do capítulo, são mostrados alguns dos resultados obtidos nos testes laboratoriais realizados com a luminária.

No capítulo 5 é abordada a implementação da luminária no piloto de testes e são mostrados igualmente alguns dos resultados obtidos.

Finalmente no capítulo 6, serão tiradas algumas conclusões ao trabalho bem como os próximos passos a tomar.

Capítulo 2

2 - Iluminação pública

2.1 - Enquadramento Histórico

Durante milhares de anos, a lua foi a única fonte luminosa existente durante o período noturno. Mais tarde, com a descoberta do fogo, para além das suas diversas utilidades, tornou-se o principal meio de iluminação. Ainda hoje, em certas civilizações menos desenvolvidas e isoladas ou em zonas mais suscetíveis a falhas de energia elétrica, o fogo é utilizado com esta finalidade.

Os primeiros candeeiros lisboetas eram chamados de lampiões de azeite, e tal como o nome indica, eram alimentados a azeite. Estes necessitavam de grande manutenção e exigiam que alguém se deslocasse a cada um deles individualmente para os acender e apagar, bem como reacender aqueles que indevidamente se apagassem.

Mais tarde em 1848, surgiu o primeiro sistema de iluminação pública a gás. Além disso, foi também o primeiro sistema de IP centralizado. O elevado custo bem como a total dependência do bom funcionamento da central de abastecimento, foram os principais problemas detetados no sistema.

Em outubro de 1878, deu-se a inauguração da rede de IP elétrica na cidade de Lisboa, quando o Rei Dom Luiz ofereceu à Câmara Municipal, seis candeeiros de lâmpadas de arco tipo *Yablochkov*[1], que tinham sido usados pela primeira vez, a 28 de setembro desse ano, na Cidadela de Cascais, por ocasião das festas de aniversário do Príncipe Dom Carlos. Estas eram iguais às que estavam instaladas na praça do Teatro da Ópera em Paris e o Rei deu ordem para a sua compra, para serem experimentadas em Portugal, que na época utilizava ainda iluminação a gás, distribuído pela *Companhia de Gaz Lisbonense*.

Após esta experiência, a novidade da instalação elétrica não se impôs logo na cidade porque a sua expansão implicava muitos custos, além de que, as relações entre a Câmara e a *Companhia de Gaz Lisbonense* não eram as melhores e estavam sujeitas a um contrato de fornecimento de gás que só viria a expirar em 1880.

Finalmente em 1891, com a criação da CRGE (Companhias Reunidas Gaz e Eletricidade), foi possível o desenvolvimento e distribuição de eletricidade em Lisboa quer na iluminação particular quer na iluminação pública, acabando a iluminação a gás por "morrer" em 1965 [5].

[1] Pavel Yablochkov foi um engenheiro elétrico russo, inventor da lâmpada Yablochkov utilizada nos primeiros sistemas de iluminação pública elétricos.

A iluminação pública continuou a evoluir até aos dias de hoje, desde as lâmpadas incandescentes ou de vapor de sódio até à tecnologia LED. Além da evolução da fonte de iluminação, as restantes partes integrantes do sistema foram sofrendo melhorias, desde o desenvolvimento de LED *drivers* de elevada eficiência até ao dotar as próprias luminárias de inteligência.

2.2 - Sistemas de Controlo

Desde o tempo em que os sistemas de iluminação pública utilizavam o fogo como fonte luminosa, muito evoluíram. Primeiramente, no tipo de fonte de iluminação utilizada, com o uso de lâmpadas elétricas em substituição ao gás e azeite, até à utilização da tecnologia LED. Além disso, foi-se tornando evidente a necessidade de dotar o sistema de inteligência.

Um sistema de controlo automático, é uma unidade que permite realizar diferentes ações pré-definidas ou programadas, dependentes da informação recolhida. No caso particular da iluminação pública, estas unidades são responsáveis por dotar o sistema de autonomia na gestão de energia. O controlo efetuado, pode ser tão simples como apenas ligar e desligar a iluminação, ou mais complexo, envolvendo a aquisição de informação e reprodução de perfis de iluminação.

2.2.1 - Arquiteturas de Controlo

Um sistema de IP pode estar organizado em diferentes topologias, cada uma delas com as suas vantagens e desvantagens e que devem ser escolhidas de acordo com o perfil do local onde este será instalado, isto é, pelo movimento expectável, tipo de local, meio circundante, entre outros. No esquema da figura seguinte, é possível ver a organização das diversas arquiteturas de controlo para sistemas de IP.

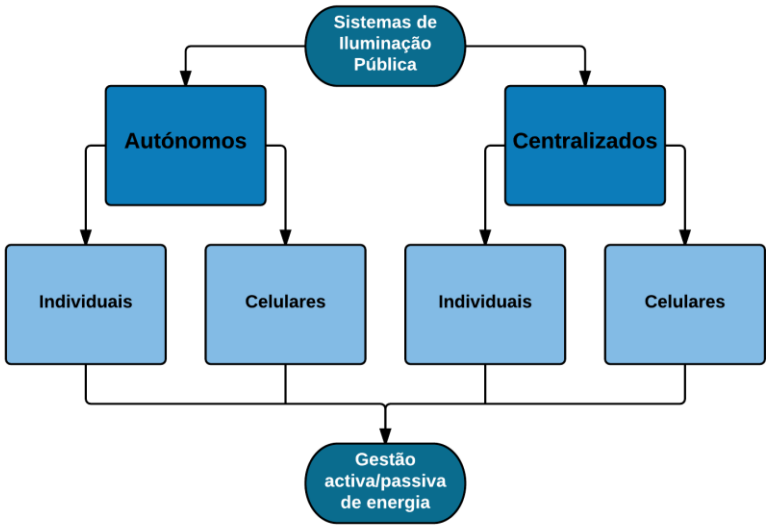


Figura 2 - Topologias dos sistemas de iluminação pública.

Da análise da figura 2, verifica-se que existem dois tipos de controlo distintos: autónomo e centralizado. Independentemente do tipo de controlo, podemos ter um princípio de funcionamento individual (*stand-alone*), onde cada unidade de controlo (UC) atua como sendo única, não havendo portanto, troca de informação com as restantes, ou celular, onde contrariamente ao *stand-alone*, a comunicação entre as várias unidades de controlo existe e estas funcionam como sendo partes (células) integrantes de um mesmo sistema.

Controlo autónomo

O controlo autónomo é uma solução de gestão, onde cada luminária possui uma unidade de controlo dedicada, permitindo assim que esta seja controlada independentemente das restantes. No caso de sistemas celulares, é possível a comunicação com o exterior, como luminárias vizinhas ou outros dispositivos de gestão remota caso esta seja suportada. O facto de cada luminária possuir um módulo de controlo torna este sistema mais robusto, mas também mais dispendioso e menos indicado para certos cenários, como centros urbanos com grande movimento, onde consequentemente, as luminárias estarão a maior parte do tempo ligadas.

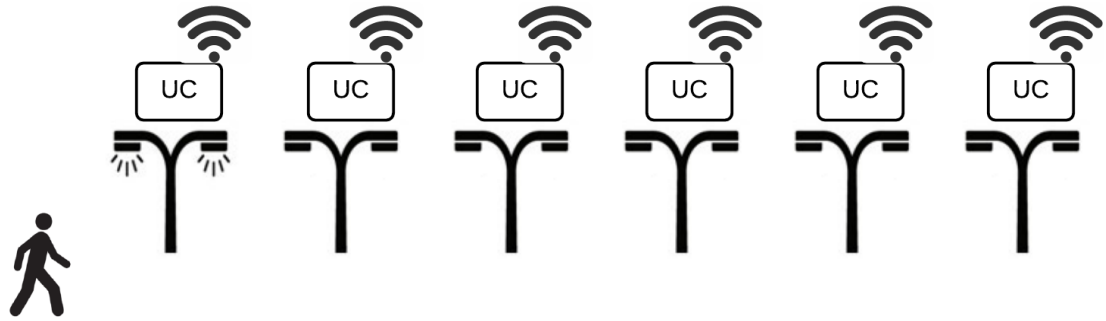


Figura 3 - Sistema de IP com topologia de controlo autónomo.

Na figura 3, pode-se ver um exemplo do funcionamento de um sistema autónomo onde foi detetado movimento pela primeira UC, que consequentemente fez ligar a luminária correspondente, permanecendo as restantes desligadas.

Controlo centralizado

Nesta topologia, a unidade de controlo é responsável por um determinado número de luminárias, que terão assim um comportamento uniforme. Este princípio de funcionamento é indicado para cenários como troços de vias públicas rodoviárias movimentadas, uma vez que, devido à sua extensão, faz todo o sentido que quando é detetado movimento, um grupo de luminárias se acenda e não apenas uma. O custo

associado a sistemas com esta arquitetura, é mais reduzido quando comparado com a anterior, perdendo no entanto robustez, uma vez que uma falha na unidade de controlo afetará todo o grupo de luminárias. Tal como no controlo autónomo, estes podem possuir ou não, a capacidade de comunicação com o exterior.

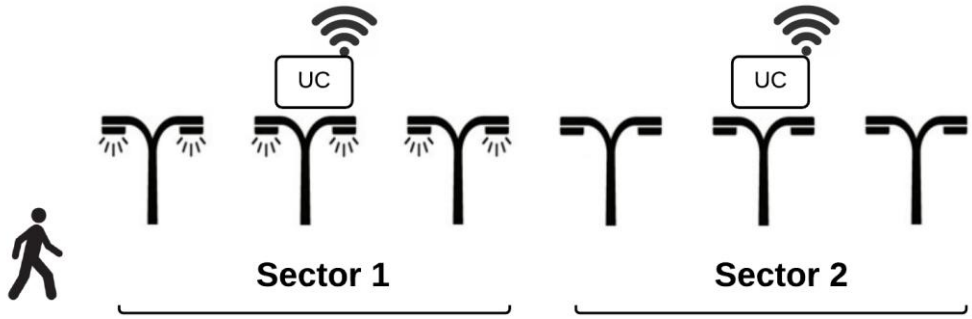


Figura 4 - Sistema de IP com controlo centralizado.

Na figura 4, é possível observar um exemplo de um sistema de IP com controlo centralizado. O movimento detetado pela UC faz ligar as luminárias de todo o setor que esta controla.

2.2.2 - Gestão energética

Uma gestão energética eficiente é, no fundo, o principal objetivo de um sistema de IP inteligente. Esta gestão pode ser passiva, como a existente nos sistemas mais tradicionais, ou ativa, utilizada em sistemas mais recentes.

Controlo Passivo

É a forma de gestão mais tradicional e simples, sendo ainda hoje a mais utilizada. A unidade de controlo liga ou desliga a iluminação de acordo com a informação obtida por sensores crepusculares ou temporizadores como relógios astronómicos. A solução com sensores crepusculares, acaba por reunir mais consenso, uma vez que permite à unidade de controlo obter informação sobre a luz natural existente, e não apenas basear-se num horário pré-definido, como quando se utilizam relógios astronómicos. Estes autómatos são utilizados para controlar a iluminação pública em função do pôr e nascer do sol, cujo horário de funcionamento se encontra enquadrado na variação do ciclo solar ao longo do ano [6]. Embora simples, esta solução de gestão poderá ter uma relação custo/benefício energético interessante quando instalada em locais com grande movimento noturno, em que as luminárias terão a necessidade de passar a maior parte do tempo ligadas na sua intensidade máxima. De notar também, que a utilização das duas solução em simultâneo, poderá trazer benefícios e deve ser tida em conta.

Controlo Ativo

O aparecimento de tecnologias de iluminação que permitem a variação do seu fluxo luminoso, abriu as portas a novas opções de gestão energética. A unidade de controlo, pode assim não só ligar e desligar a iluminação, como fazer variar o fluxo luminoso, dependendo da informação recolhida pelos sensores. Torna-se assim possível, por exemplo, que em zonas abrangidas por outras fontes de iluminação circundantes, não seja necessário ligar uma ou várias luminárias na sua intensidade máxima, mas sim num nível suficiente de iluminação. Em cenário oposto, na inexistência de qualquer movimento ou luz, seja esta natural ou artificial, em vez da escuridão, se possa ter um certo nível de iluminação, dando mais comodidade e segurança às pessoas, e sem um gasto energético considerável.

2.2.3 - Topologias de Sistemas de Iluminação Pública

Existem diversas topologias que se podem utilizar na instalação de um sistema de IP, cada uma com as suas vantagens e desvantagens. A escolha da topologia que melhor se adequa ao perfil do local de instalação é extremamente importante para a obtenção da melhor relação custo/benefício. De seguida, serão demonstradas e explicadas algumas das principais topologias utilizadas.

Sistema autónomo, individual e com controlo passivo de energia

É a topologia mais antiga e mais utilizada nos sistemas de IP já dotados com algum tipo de controlo. Como sistema individual, cada luminária é independente das restantes, não havendo qualquer troca de informação entre elas. Neste caso, o controlo resume-se a ligar e desligar a iluminação, seja pela ausência de luz natural (quando utilizados sensores crepusculares) ou por nos encontrarmos em determinada hora do dia (quando utilizados relógios astronómicos). Na figura seguinte, pode-se observar um diagrama de fluxo representativo deste tipo de solução.

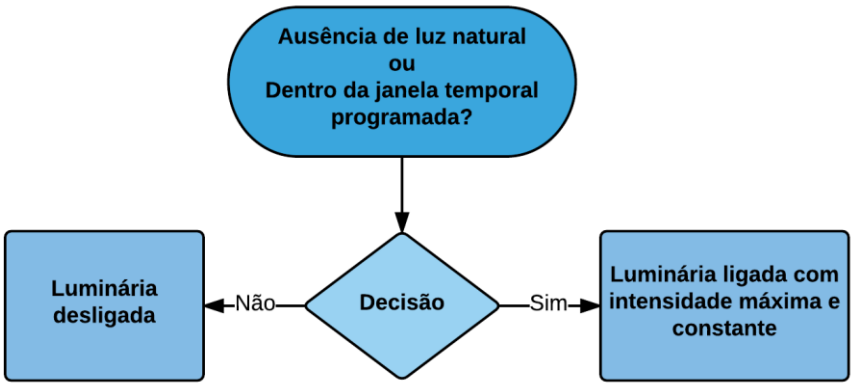


Figura 5 - Diagrama de fluxo de um sistema de IP autónomo, individual e com gestão passiva de energia.

Sistema autónomo, celular e com controlo ativo de energia

É uma opção muito interessante para locais rurais, onde existe pouco movimento durante o período noturno. Nesta topologia, o sistema tem a capacidade de controlar o fluxo luminoso e a comunicação entre luminárias vizinhas existe, possibilitando aumentar a eficiência energética do sistema.

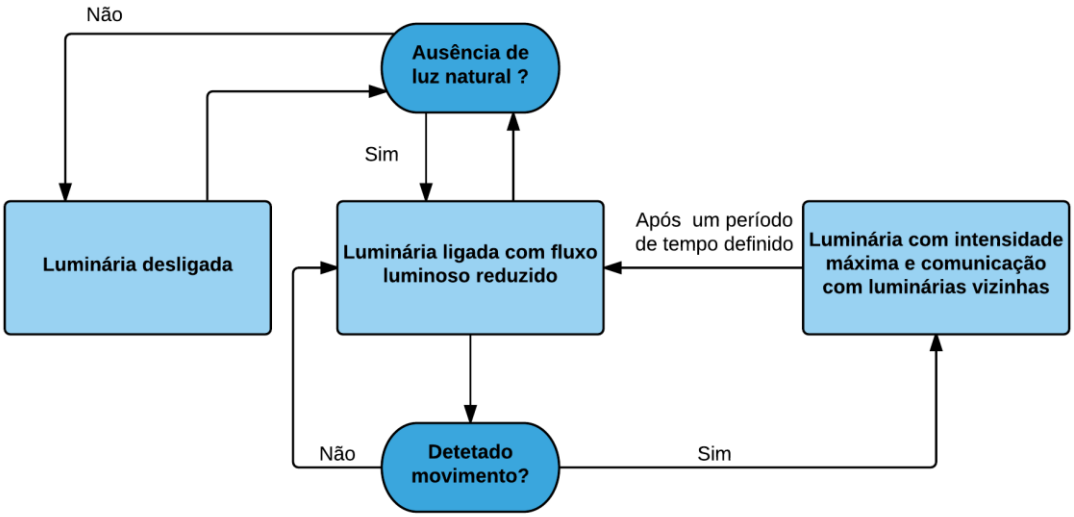


Figura 6 - Diagrama de fluxo de um sistema de IP autónomo, celular e com controlo ativo de energia.

Sistema centralizado, individual e com gestão passiva de energia

É uma topologia em que o sistema não é dotado da capacidade de controlar o fluxo luminoso. Cada unidade de controlo tem associado um número variável de luminárias, sendo responsável por as ligar e desligar quando necessário.

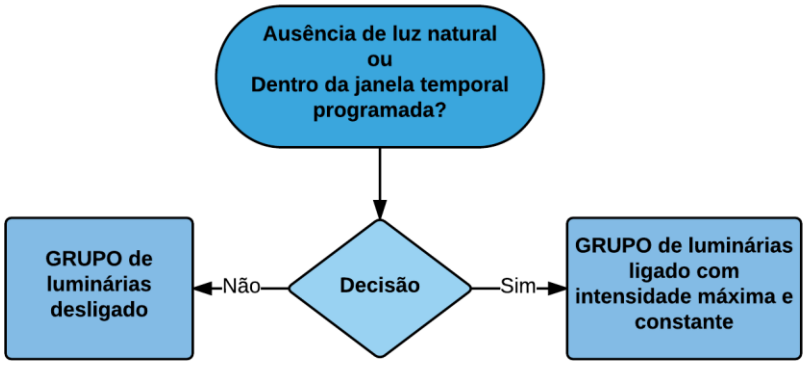


Figura 7 - Diagrama de fluxo de um sistema de IP centralizado, individual e com gestão passiva de energia.

Sistema centralizado, celular e com gestão ativa de energia

Nesta topologia, e tal como na anterior, cada unidade de controlo é responsável por um determinado grupo de luminárias. No entanto, esta solução permite um controlo inteligente do fluxo luminoso bem como a comunicação com unidades de controlo na vizinhança. A figura 8 é representativa desta solução de controlo.

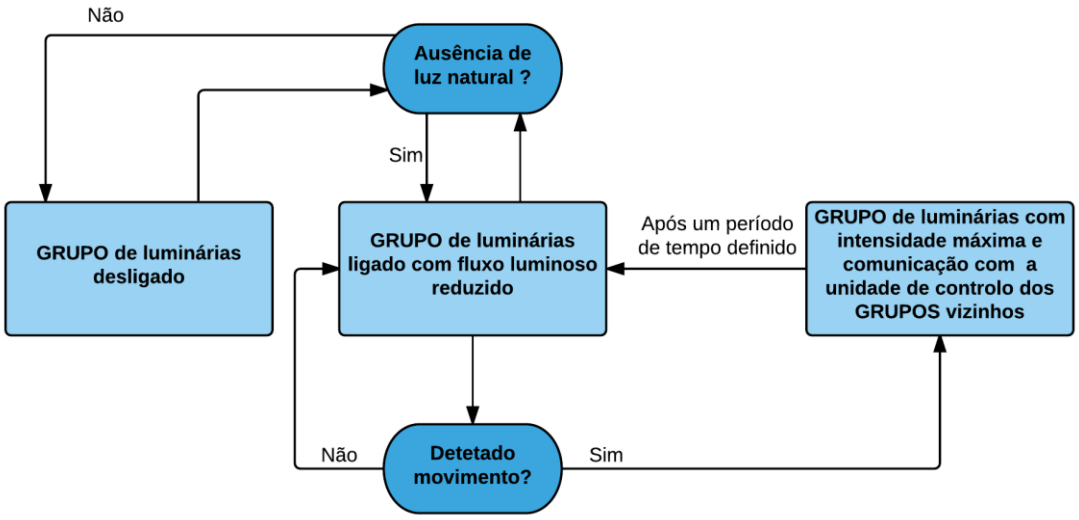


Figura 8 - Diagrama de fluxo de um sistema de IP centralizado, celular e com gestão ativa de energia.

2.3 – Resumo e observações

Neste capítulo foram abordadas as diferentes topologias dos sistemas de IP, assim como os principais métodos de gestão energética utilizados. Uma correta escolha da topologia e do método de gestão que melhor se adequa ao local, é imprescindível para a obtenção da melhor relação custo/benefício possível. Os sistemas autónomos, onde cada luminária possui uma unidade de controlo dedicada, são os que apresentam uma maior robustez e versatilidade, no entanto, implicam um custo mais elevado. Nos sistemas centralizados, cada unidade de controlo é responsável por um determinado grupo de luminárias, resultando numa menor robustez (uma falha na unidade de controlo compromete um número de luminárias que pode ser considerável), mas a um custo mais acessível. Independentemente de serem autónomos ou centralizados, estes podem possuir comunicação (celulares) ou não (individuais) com o exterior, assim como um controlo de gestão energética ativo (onde é possível controlar o fluxo luminoso) ou passivo (quando o controlo se resume a ligar e desligar a luminária).

Um estudo cuidadoso do local de instalação do sistema, torna-se assim muito importante. Optar por uma solução de controlo mais exigente financeiramente em zonas de grande movimento noturno, onde a iluminação estará praticamente o tempo todo ligada, poderá não se justificar, assim como a utilização de um sistema autónomo em

vias extensas, poderá trazer custos adicionais desnecessários. Os sistemas dotados de capacidade de comunicação podem permitir não só a comunicação com luminárias vizinhas, como também gestão remota. Esta funcionalidade será abordada com maior detalhe ao longo da dissertação.

Capítulo 3

3 - Sistema de Iluminação Pública Inteligente

Na figura 9 é possível observar um diagrama de blocos simplificado de um sistema de iluminação pública inteligente com gestão remota.

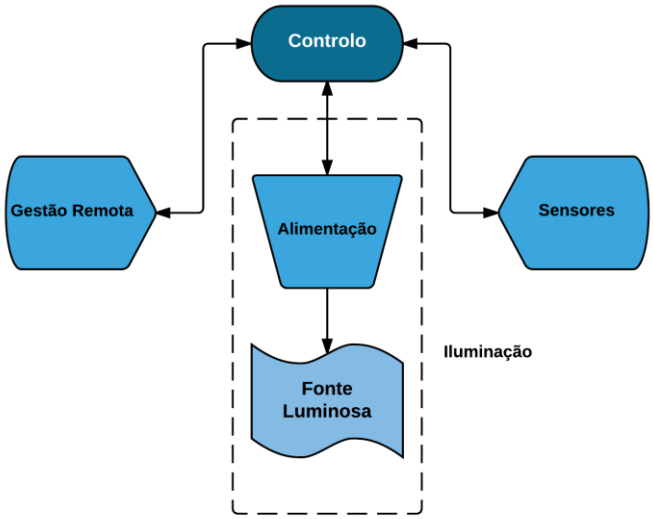


Figura 9 - Diagrama de blocos de um sistema de IP inteligente.

Quando se fala num sistema inteligente, o uso de sensores é indispensável uma vez que são estes que permitem recolher a informação necessária para que a unidade de controlo atue de acordo com as necessidades. A comunicação torna-se assim indispensável para que toda a informação seja trocada entre as devidas partes do sistema.

A gestão remota pode conter monitorização do sistema (em tempo real ou não), onde é possível observar os dados recolhidos pelos sensores ou estados de iluminação por exemplo. Ainda neste bloco, podemos ter a possibilidade de configuração/reconfiguração do sistema (remotamente), havendo a possibilidade de alteração de determinados parâmetros do sistema em função dos dados recolhidos, otimizando assim o funcionamento global do sistema.

O bloco de alimentação é o responsável por fornecer energia à fonte luminosa utilizada e é fundamental que este permita de alguma forma ser controlado externamente pela unidade de controlo. Tanto a fonte de alimentação, como os dispositivos de iluminação, desempenham um papel fundamental, uma vez que se estes não forem energeticamente eficientes, o principal objetivo do sistema fica comprometido.

3.1 - Sensores

Na sua essência, um sensor é um elemento transdutor[2] que reage a alterações ou estímulos, físicos ou químicos, ocorridos no seu meio envolvente, de uma forma determinística e mensurável.

Hoje em dia, pode-se encontrar uma grande variedade de sensores que nos permitem obter informação sobre quase tudo o que nos rodeia, desde luz, cor, som, pressão, temperatura, humidade ou movimento por exemplo. Os mesmos estão presentes nos mais diversos dispositivos eletrónicos que utilizamos no dia-a-dia.

Neste caso em particular, tratando-se de um sistema de iluminação pública, as informações mais relevantes que pretendemos obter estão relacionadas com a deteção de movimento, a existência de iluminação natural, as condições climatéricas do meio e as condições a que o *hardware* do sistema está sujeito.

Desta forma, podem-se dividir as variáveis sensoriais do sistema em três grupos como mostra a figura 10.

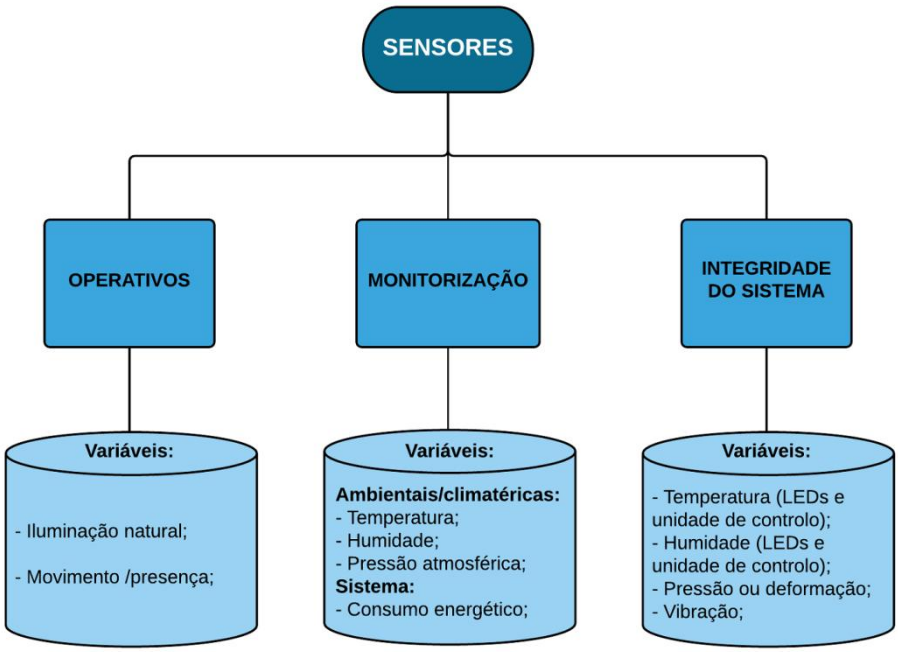


Figura 10 - Sensores nos sistemas de IP inteligente.

Os sensores operativos são aqueles que interferem diretamente no modo de operação do sistema uma vez que irão dar a informação necessária à UC, para que esta possa controlar o fluxo luminoso, seja pela deteção ou não de movimento ou pela quantidade de luz natural existente.

Em sistemas que permitam gestão remota, variáveis como temperatura, humidade e consumo energético têm grande importância para a gestão do sistema, pois indicam as

[2] Transdutor é um dispositivo que transforma um tipo de energia em outro.

condições a que o *hardware* está submetido. Os dados climatéricos sobre o local onde o sistema está instalado, podem também ter uma importância relevante, como por exemplo para estudos climatéricos da região ou correlação entre consumos e variações ambientais.

Por fim, é também importante obter informações sobre a integridade do sistema. Temperaturas elevadas afetam não só a qualidade da luz emitida pelos LED's como reduzem o seu tempo de vida. Além da temperatura, a humidade é também um fator importante para a integridade quer dos LED's quer da unidade de controlo, uma vez que em excesso poderá causar danos. Vibrações excessivas ou algum tipo de pressão ou deformação sofrida poderá igualmente comprometer o bom funcionamento do sistema. Estas podem ser causadas não só por atos de vandalismo, mas também por causas naturais como ventos e chuvas fortes.

A quantidade e diversidade de sensores existentes são gigantescas, de tal forma que de seguida, será feito um estudo de alguns sensores de interesse para este sistema.

3.1.1 - Sensores de temperatura

A temperatura pode desempenhar um papel importantíssimo no bom funcionamento do sistema. Além dos sensores poderem fornecer informações sobre as condições onde o *hardware* está inserido, especialmente sobre a temperatura dos LED's, podem também desempenhar funções de monitorização climatérica se aplicados num local exterior da luminária. Há no entanto que ter em conta que, num sistema de IP, teremos um número considerável de luminárias relativamente próximas umas das outras, pelo que a integração destes sensores não será necessário em todas elas uma vez que apenas originaria informação redundante.

Termopar

É dos sensores mais utilizados muito devido ao seu baixo custo, elevada gama de temperaturas e fácil substituição. Um termopar é constituído pela junção de dois metais diferentes que quando aquecida gera uma corrente elétrica. Este fenómeno é denominado por efeito de Seebeck e está exemplificado na figura 11 a). Quando o circuito é aberto na extremidade não aquecida, surge uma força eletromotriz, como se pode ver através da figura 11 b).

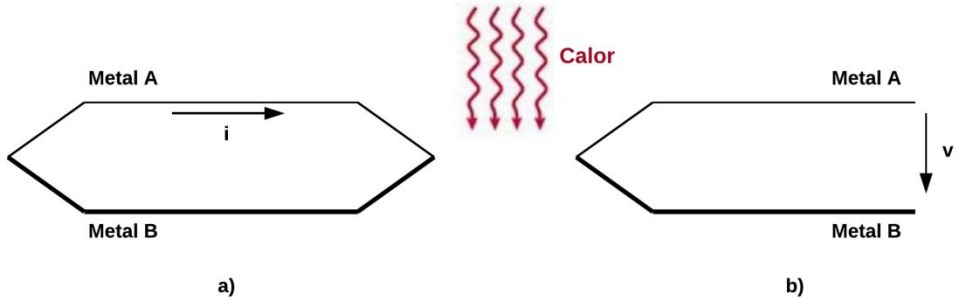


Figura 11 - Efeito de Seebeck [a)] e geração de tensão de Seebeck (termopar) [b)]. [8].

Termorresistência (RTD)

As termorresistências, sensores de resistência térmica, ou ainda RTD's (do inglês, *Resistance Temperature Detector*), baseiam-se no princípio da dependência da resistividade de um material condutor com a temperatura. Tal como os termopares, também estes possuem uma elevada amplitude térmica, embora não atinjam temperaturas tão elevadas. Sendo dispositivos passivos, os RTD's necessitam da injeção de corrente, que irá resultar num valor de tensão mensurável.

Matematicamente, o seu comportamento pode ser caracterizado através de polinómios que podem ter várias ordens. Ordens mais baixas são computacionalmente mais simples, mas esta terá menos exatidão. Contrariamente, a utilização de polinómios de ordem mais elevada, aumenta a complexidade, mas são mais exatos. Uma primeira aproximação pode ser feita através da seguinte equação:

$$R_T = R_0.[1 + \alpha(T - T_0)]\ [\Omega] \tag{3.1}$$

Onde R_T é o valor da resistência à temperatura T, R_0 o valor da resistência à temperatura T_0 e α o coeficiente de temperatura do sensor.

Termístor

Os termístores são sensores de temperatura, baseados em semicondutores. São normalmente constituídos por materiais cerâmicos ou polímeros ao contrário dos RTD's e dos termopares, que são compostos por metais. Estes possuem menor amplitude térmica quando comparados com os anteriores, mas possuem maior precisão. Podem-se encontrar dois tipos de termístores que funcionam de forma inversa, os PTC's (do inglês, *Positive Temperature Coefficient*) cuja resistência aumenta com o aumento da temperatura e os NTC's (do inglês, *Negative Temperature Coefficient*) onde a resistência diminui com o aumento da temperatura. A caracterização destes sensores pode ser efetuada através da expressão:

$$R_T = R_0.e^{\beta\left(\frac{1}{T} - \frac{1}{T_0}\right)}\ [\Omega] \tag{3.2}$$

Onde R_T é o valor da resistência à temperatura T, R_0 o valor da resistência à temperatura T_0 e β um parâmetro constante do sensor.

Sensores de temperatura Integrados

Este tipo de sensores são baseados em circuitos eletrónicos que podem comportar-se como fontes de corrente ou como reguladores de tensão, e onde as unidades de corrente e tensão respetivamente, são diretamente proporcionais à variação de temperatura. Estes apresentam em geral, uma excelente linearidade, mas estão restringidos a uma gama de temperaturas mais limitada. Também é possível encontrar

sensores deste tipo com saída digital, onde o valor a medir (de temperatura neste caso) é transmitido por meio de um protocolo de comunicação como SPI[3], I^2C [4]ou 1-wire[5].

3.1.2 - Sensores de humidade

A humidade tem também bastante interesse para o sistema, quer dentro da luminária, para ser monitorizada e não permitindo que atinja níveis prejudiciais para o sistema, ou fora como variável climatérica do local onde o sistema está instalado.

Pode ser definida como a quantidade de vapor de água na atmosfera. É geralmente utilizado o valor de humidade relativa, que é definido como a razão da quantidade de vapor de água presente numa porção de atmosfera, com a quantidade máxima de vapor de água que a atmosfera pode suportar a uma determinada temperatura. É possível encontrar no mercado os dois tipos: os sensores de humidade absoluta e os sensores de humidade relativa.

Os sensores de humidade eletrónicos são os mais fáceis de encontrar e utilizar, podendo ser divididos em dois tipos baseados no princípio sensorial, os resistivos e os capacitivos.

3.1.3 – Sensores de pressão atmosférica

A pressão atmosférica ou pressão barométrica é a força normal exercida pelo peso do ar numa superfície, por unidade de área. A expressão matemática é dada por:

$$p = \frac{F}{A} \ [N/m^2] \tag{3.3}$$

Em que p é o valor de pressão, F a força normal e A a área da superfície de contacto. Essa força é geralmente expressa em *Pascal* (Pa), atmosfera (atm) ou bar (bar), sendo $1\ Pa = 1 \times 10^{-5} bar$ e $1\ atm = 101\ 325\ Pa$.

O valor da pressão atmosférica é um dado que pode ter interesse para monitorização e estudos climatéricos do local. Dado que a pressão atmosférica varia com a altitude, é possível estabelecer uma relação entre ambas e obter o seu valor através da expressão barométrica:

$$p = p_0 e^{-\frac{h}{h_0}} \ [Pa] \tag{3.4}$$

[3]*Serial Peripheral Interface* (SPI) é um protocolo série, *Single-Master Multi-Slave* e síncrono que utiliza quatro linhas, para comunicações de curtas distâncias (geralmente em sistemas embebidos).

[4]*Inter-Integrated Circuit* (I^2C), é um protocolo série, *Multi-Master Multi-Slave*, que utiliza apenas duas linhas para comunicações a curtas distâncias e mais lentas.

[5]*1-wire* é um protocolo conceptualmente similar aoI^2C, que utiliza apenas um condutor para comunicação, embora a taxas de transferência mais baixas.

Onde p é a pressão atmosférica (Pa), p_0 a pressão atmosférica ao nível do mar (Pa), h a altitude (m) e h_0 o valor da altura de escala da atmosfera (m).

Sabendo-se que a pressão atmosférica ao nível do mar é $1\ atm$ e que o valor da altura de escala da atmosfera é aproximadamente $8\ km$, é possível obter uma expressão que relaciona a altitude com a pressão atmosférica e onde se pode verificar que esta diminui exponencialmente com a altitude. Desta forma, com a utilização de um sensor de pressão é possível obter informação sobre dois fatores climatéricos distintos.

3.1.4 - Sensores de vibração

Sensores de *Tilt*

Os sensores de *tilt* são dispositivos pequenos, baratos, de baixo consumo e fáceis de utilizar, que permitem detetar a orientação e inclinação do objeto em que estão inseridos. Geralmente são constituídos por uma cavidade com duas esferas de material condutor. Uma das extremidades do sensor contém dois elementos (ou pólos) condutores que ficam em curto-circuito (devido às esferas) sempre que o sensor ultrapasse uma determinada inclinação.

Acelerómetros

Os acelerómetros são muito mais do que sensores de vibração. São dispositivos sensíveis a forças estáticas (gravidade) e dinâmicas (movimento/vibração) de aceleração. O exemplo mais comum para explicar o seu funcionamento, é imaginar um acelerómetro em repouso na superfície da Terra. Nesta situação, o valor medido seria o valor da aceleração gravítica ($g = 9.8\ m/s^2$) e não 0. O mesmo dispositivo, em queda livre em direção ao centro da Terra, a uma taxa de aceleração de $9.8 m/s^2$ medirá 0.

Podem-se encontrar acelerómetros com capacidade de medição em 1, 2 ou 3 eixos (x, y, z), com saída analógica, digital (através de protocolos como I^2C ou SPI) ou por Modulação por Largura de Pulso, mais conhecida por PWM (do inglês, *Pulse Width Modulation*).

Na figura seguinte, é possível observar a representação de um acelerómetro de 3 eixos.

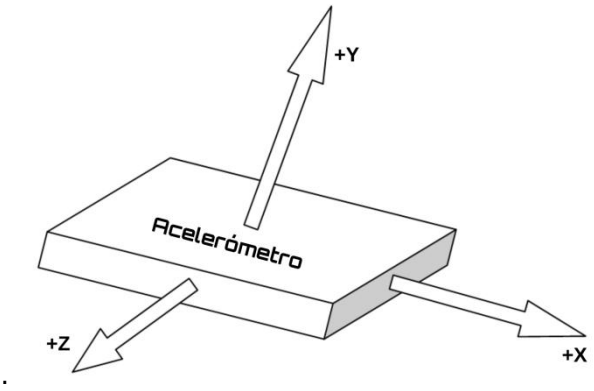


Figura 12 - Representação de um acelerómetro de 3 eixos.

- **Giroscópio**

São sensores sensíveis a movimentos de rotação e alterações na sua orientação. Tal como os acelerómetros são muito mais do que sensores de vibração, estando ambos presentes nos mais diversos dispositivos eletrónicos utilizados hoje em dia, desde telemóveis ou máquinas fotográficas, carros e naves espaciais. Podem utilizar transdutores piezoelétricos como cristais ou materiais cerâmicos ou de silício.

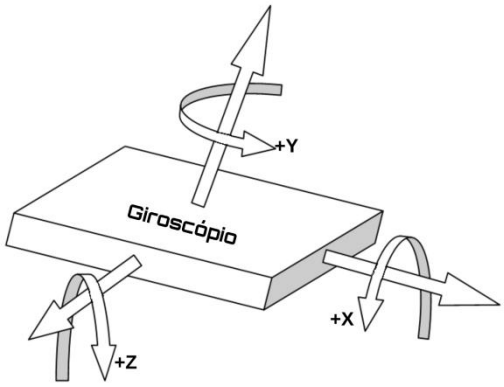


Figura 13 - Representação de um giroscópio.

3.1.5 - Sensores de movimento/presença

Estes sensores desempenham um papel fundamental no bom funcionamento dos sistemas de IP. Uma vez que a principal forma de poupar energia é impedindo que a iluminação esteja ativa quando não é necessário, isto é, quando não há movimento de pessoas/automóveis. Estes sensores serão os responsáveis por dar essa informação à unidade de controlo. Há diversas formas de detetar movimento e será feito um estudo dos principais sensores e/ou tecnologias para esta aplicação.

Sensores de pressão

Estes sensores permitem a deteção de forças ou deformações aplicadas numa área, através do uso de diafragmas ou pistões por exemplo. Podem ser piezoresistivos ou piezoelétricos, caso a resposta à deformação seja na variação da sua resistividade ou carga elétrica respetivamente. Também podem ser capacitivos ou eletromagnéticos como é o caso dos LVDT (*Linear Variable Differential Transformer*). A grande desvantagem destes sensores neste tipo de aplicação, além do custo mais elevado, é a sua difícil e específica aplicação no pavimento.

Detetores de movimento com vídeo

Embora o custo das câmaras de vídeo digitais seja cada vez menor, esta solução implicaria um elevado processamento de sinal que aumentaria o custo total desta

solução. Seria no entanto interessante, caso se pretendesse a gravação e/ou monitorização dos movimentos, embora por questões legais de privacidade, a sua aplicação em espaços públicos seja extremamente complicada, podendo mesmo inviabilizar esta solução.

Sensores optoeletrónicos

Estes sensores convertem radiação eletromagnética em sinais elétricos. Os mais utilizados fazem uso da gama do infravermelho.

A radiação infravermelha ($700nm - 1mm$), como é possível observar na figura 14, situa-se no espetro eletromagnético entre a gama visível e a gama das micro-ondas.

Toda a matéria com temperatura acima do zero absoluto emite radiação térmica, sendo a luz visível e infravermelha exemplos dessa radiação. A radiação infravermelha emitida por um objeto depende de vários fatores como temperatura, cor e textura.

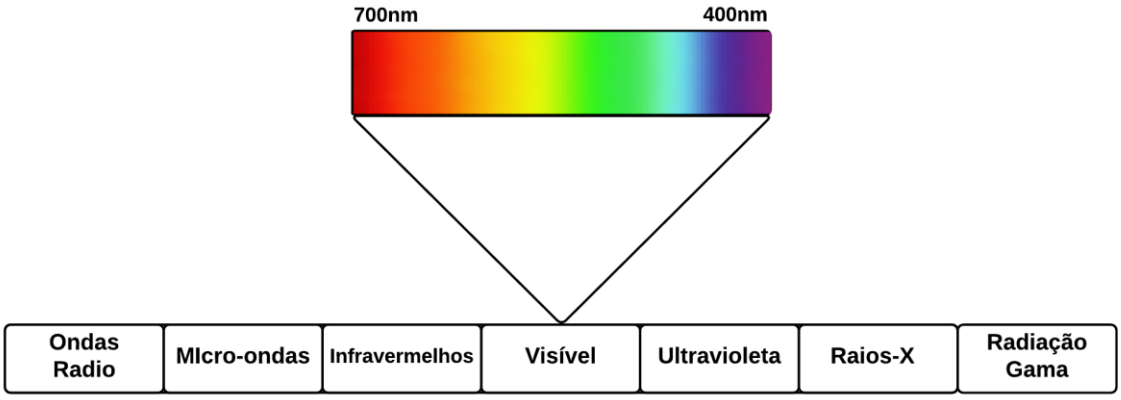


Figura 14 - Representação do espectro eletromagnético.

Este princípio de funcionamento torna-os numa boa opção para serem utilizados como sensores de movimento. Uma vez que a maior parte da radiação térmica emitida por um objeto perto da temperatura ambiente é infravermelha, é possível que qualquer ser humano bem como alguns animais e veículos motorizados sejam detetados pelo sensor.

Podem-se encontrar sensores optoelectrónicos que são imunes a variações de temperatura, mas que têm um consumo relativamente elevado. Os passivos são sensíveis a variações de temperatura, mas têm um consumo significativamente mais baixo, pois não emitem qualquer energia para a deteção.

Sensor PIR

Um PIR (do inglês, *Passive Infrared sensor*) é um sensor que tem a capacidade de captar a radiação infravermelha existente dentro do seu campo de visão (FoV, do inglês *Field of View*).

Estes sensores podem possuir um, dois ou até mesmo quatro elementos (entradas) sensíveis ao infravermelho. Os mais comuns, possuem dois elementos, e quando a radiação captada por elas é diferente, indica que foi detetado movimento. Na figura 15 b) é possível observar a representação estrutural de um PIR com dois elementos sensoriais.

Percebe-se assim que estes sensores, apenas são capazes de detetar o movimento de um objeto e não a sua presença, o que pode ser limitativo em algumas aplicações.

Quando radiação infravermelha atinge a superfície do sensor o potencial resultante é aplicado num transístor do tipo FET (*Field Effect Transistor*) gerando uma corrente elétrica. Sendo o sensor sensível a uma elevada gama de frequências de radiação, para otimizar a deteção de seres humanos, é inserido um filtro para limitar a radiação absorvida entre os $8\,\mu m$ e os $14\,\mu m$, sendo que tipicamente um ser humano emite radiação em torno dos $10\,\mu m$ [9].

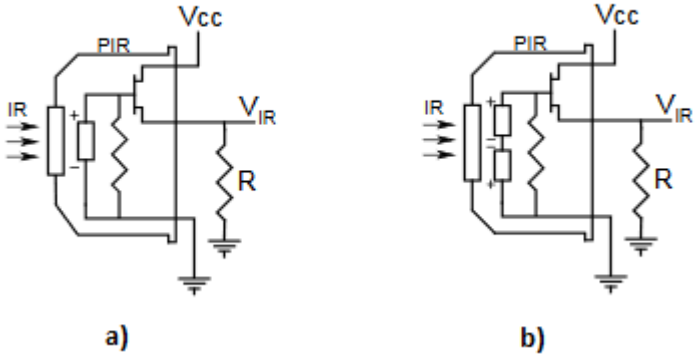


Figura 15 - Exemplo estrutural de um PIR com uma entrada [a)] e com duas entradas [b)] [9].

De forma a aumentar o FoV do sensor, é geralmente utilizado em conjunto com este uma lente de Fresnel. Inventada pelo físico francês Augustin-Jean Fresnel, foi concebida originalmente para uso em faróis de sinalização marítima. O seu desenho permite criar lentes com grande abertura e curta distância focal sem o peso e volume de material que seriam necessários a uma lente convencional.

Sensores de infravermelhos ativos

Ao contrário dos anteriores, estes sensores emitem radiação, permitindo não só detetar movimento, como também presença. São constituídos por dois elementos, um emissor, onde é geralmente utilizado um LED ou Laser, e um recetor, que pode ser constituído por um foto-díodo ou foto-transístor, que podem estar ou não instalados no

mesmo plano. Nas figuras 16 e 17 pode-se observar dois exemplos do funcionamento distinto destes sensores.

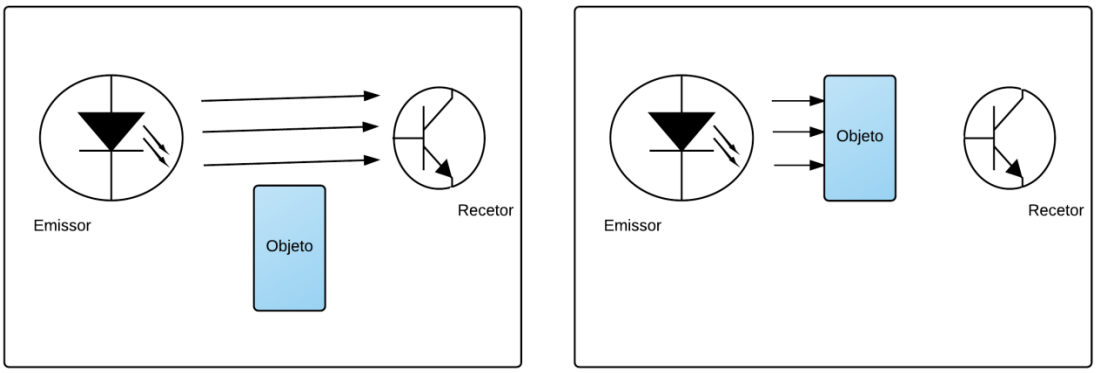


Figura 16 - Exemplo do princípio de funcionamento de um sensor ativo por transmissão.

No caso da figura 16, o emissor e o recetor estão em planos diferentes e alinhados. Quando não existe movimento, o feixe emitido pelo emissor é recebido pelo recetor. A deteção é feita, quando algum objeto interromper o feixe e consequentemente o recetor o deixar de receber a radiação emitida pelo emissor.

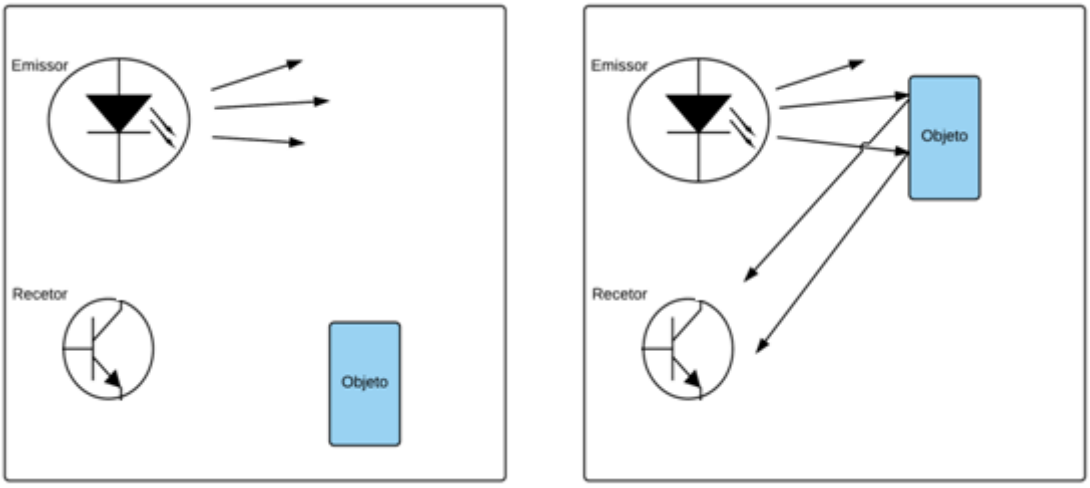


Figura 17 - Exemplo do princípio de funcionamento de um sensor ativo por reflexão difusa.

Na figura 17, é demonstrado um exemplo completamente diferente ao anterior. Neste caso, o emissor e o recetor estão instalados no mesmo plano e geralmente sob o mesmo encapsulamento. A radiação emitida é refletida para o recetor pelo objeto intruso, e é desta forma feita a deteção.

Sensores de micro-ondas

São sensores que fazem uso da gama das micro-ondas para detetar movimento. É possível encontrar sensores passivos, que de forma idêntica aos sensores optoeletrónicos passivos, apenas captam a radiação (neste caso micro-ondas) emitida pelos objetos no seu campo de visão. Os sensores ativos são compostos por um emissor e um recetor. Funcionam através da emissão (continua ou pulsada) de ondas que ao serem refletidas, são captadas pelo recetor. Devido ao efeito de Doppler[6], se as ondas transmitidas embaterem num objeto em movimento, serão refletidas e captadas pelo recetor apresentando um desvio em frequência em relação à onda transmitida, sendo assim detectado movimento.

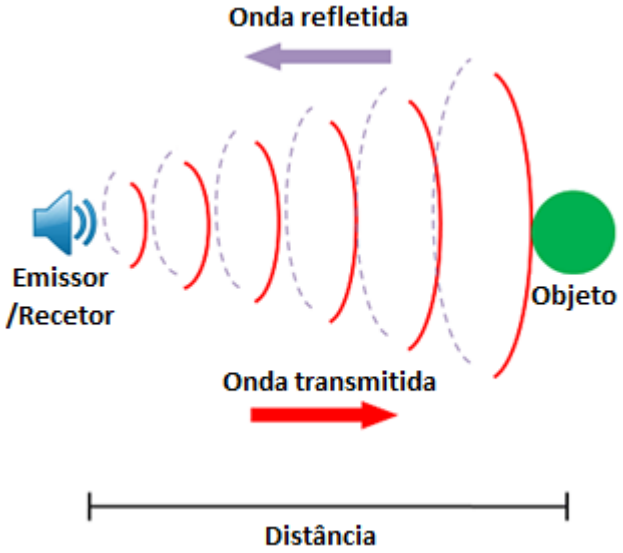


Figura 18 - Princípio de operação de um sensor de micro-ondas.

Sensores de ultrassons

O seu princípio de funcionamento é similar aos sensores de micro-ondas, na medida em que são emitidas ondas que após serem refletidas são captadas pelo recetor fazendo igualmente uso do efeito de *Doppler* para a deteção de movimento. Neste caso as ondas estão situadas na gama dos ultrassons e não nas micro-ondas. Geralmente os transdutores de ultrassons utilizam cristais piezoelétricos que entram em ressonância com a frequência desejada, convertendo energia elétrica em acústica e vice-versa. Ao contrário dos sensores de micro-ondas, estes estão mais suscetíveis às condições atmosféricas, nomeadamente o nevoeiro e a chuva, para além de interferirem com alguns animais que possuam a capacidade de ouvir som nesta gama de frequências.

[6] O *efeito de Doppler* é um fenómeno físico que consiste na alteração da frequência vista por um observador, quando a fonte emissora está em movimento.

Sensores híbridos

Estes sensores são assim denominados, pois utilizam mais do que uma tecnologia na tentativa de tirar partido das melhores características de cada uma. Utilizam geralmente um sensor passivo, como o PIR por exemplo, que devido ao seu baixo consumo, “trabalha” a maior parte do tempo. No entanto, este sensor é muito suscetível a falsos positivos causados por bolsas de ar quente e poderá ter dificuldade em detetar movimento em dias mais quentes. Combinado com o sensor passivo, é geralmente utilizado um sensor de micro-ondas ou de ultrassons, que face à sua imunidade a variações de temperatura, torna-se um excelente complemento ao PIR, fornecendo assim uma maior robustez ao sensor híbrido.

3.1.6 - Sensores de corrente e tensão

São importantes pois possibilitam monitorizar o consumo do sistema e até mesmo a sua integridade, no caso da leitura de valores anormais.

Existem várias soluções para a obtenção desta informação. Algumas são mais invasivas, que implicam a alteração do circuito projetado, para a adição de um circuito adicional, seja este integrado ou com recurso a componentes discretos. Há no entanto, a possibilidade de utilizar sensores que em nada interferem com o circuito projetado, como é o exemplo dos sensores de efeito de Hall para a medição de corrente elétrica.

Sensores de efeito de Hall

A corrente elétrica pode ser definida como o deslocamento de cargas dentro de um condutor, quando existe uma diferença de potencial nas suas extremidades. Geralmente o sensor de Hall, é construído por uma folha fina de material condutor que quando sujeita a um campo magnético responde com uma diferença de potencial (ou tensão de Hall) perpendicular ao sentido da corrente, como se pode observar no esquema da figura 19.

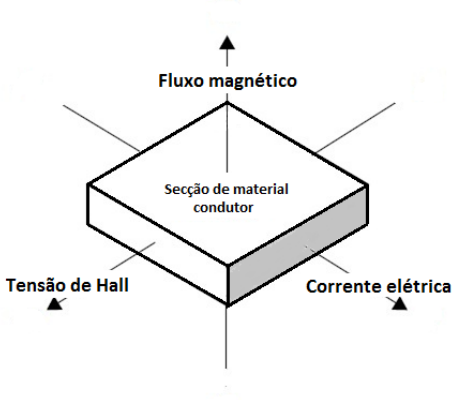


Figura 19 - Ilustração do princípio do efeito de Hall.

3.1.7 - Sensores de luz

A iluminação ou luz natural existente, é um dado muito importante para este sistema. Para além de evitar que a iluminação artificial esteja ligada quando não há movimento, pode impedir que se ligue a iluminação artificial quando a luz natural existente é suficiente. Segue-se um estudo de alguns sensores que nos permitem obter essa informação.

Foto-díodos

É tal como o nome indica, um díodo fotossensível, isto é, sensível à luz. Este componente é semicondutor e tal como os restantes díodos semicondutores é constituído por uma junção p-n ou p-i-n[7], sendo que no caso dos foto-díodos, o encapsulamento transparente permite que a luz incida sobre ela, gerando pares eletrão-lacuna que quando sobre o efeito de um campo elétrico externo pode conduzir corrente. Podem ser usados com polarização direta, onde ocorre um aumento exponencial da corrente elétrica, ou inversa, que apresenta uma maior linearidade, sendo por isso, a mais utilizada. A principal desvantagem destes sensores é o baixo sinal de saída que requer amplificação.

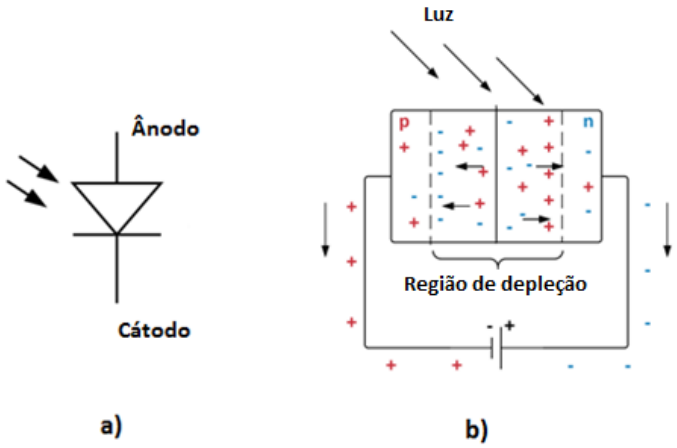


Figura 20 – Representação [a)] e exemplo estrutural [b)] de um foto-díodo.

Foto-transístor

Tem um princípio de funcionamento idêntico ao dos foto-díodos. Neste dispositivo, os fotões incidem sobre a junção p-n da base-coletor de um transístor, geralmente bipolar do tipo-n. Pode-se assim dizer que no fundo, um foto-transístor é um foto-díodo com amplificação, e como tal, mais sensível à luz, embora com um tempo de resposta mais elevado que o foto-díodo.

[7]Uma junção p-i-n difere da junção p-n, na existência de uma zona intrínseca não dopada, entre as junções p e n, alterando assim algumas das suas propriedades.

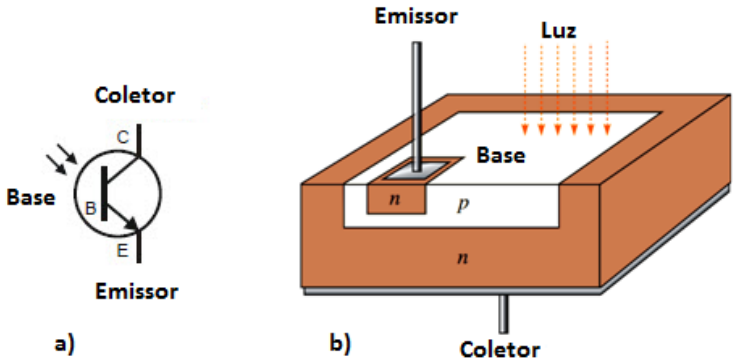


Figura 21 – Representação [a)] e exemplo estrutural [b)] de um foto-transístor.

Foto-resistências

Estes sensores geralmente denominados de LDR's (do inglês, *Light Dependent Resistor*), são componentes passivos onde tal como o nome indica, a sua resistência elétrica depende da luz incidente. São construídos a partir de material semicondutor fotossensível como o Sulforeto de Cádmio (CdS) ou o Seleneto de Cádmio (CdSe). Estes possuem uma resistência elevada, que diminui com o aumento da luz incidente. Este fenómeno ocorre, porque quando a luz incide sobre o material, os fotões libertam eletrões, aumentando assim a condutividade e consequentemente diminuindo a sua resistividade. Os LDR's apresentam a vantagem de terem um custo mais baixo comparado com os anteriores e uma maior sensibilidade, no entanto, são muito sensíveis às variações de temperatura e possuem um tempo de resposta mais elevado.

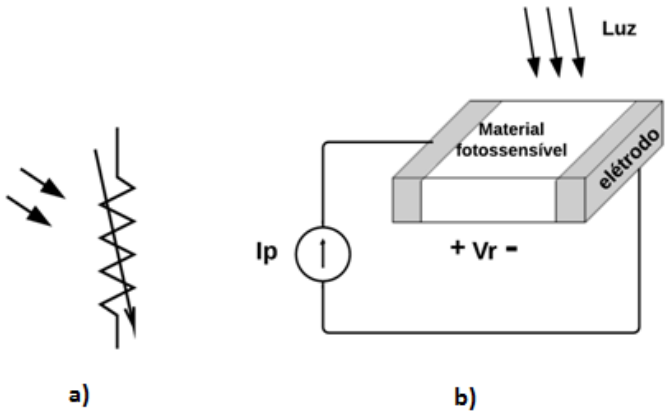


Figura 22 – Representação [a)] e exemplo estrutural [b)] de uma foto-resistência [7].

3.2 – Gestão Remota

Um sistema com gestão remota pode permitir ao gestor do sistema a monitorização de dados relativos a consumos energéticos, condições atmosféricas ou fluxos de movimento por exemplo. Além disso, e mediante a informação recolhida e monitorizada, pode também, caso seja necessário, alterar certos parâmetros do sistema para melhorar o funcionamento e eficiência deste.

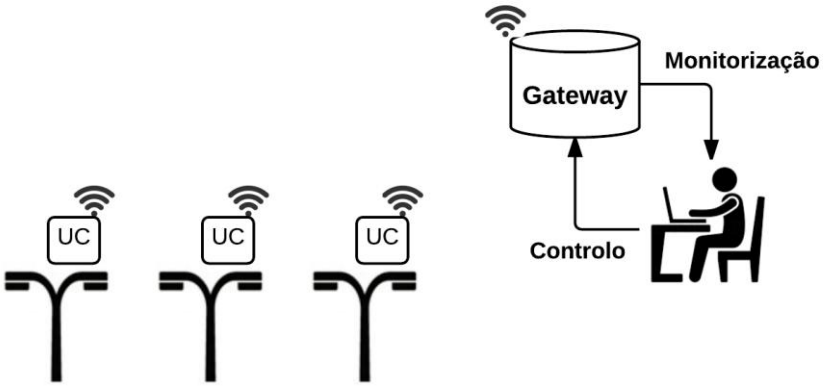


Figura 23 - Representação de um sistema de IP com gestão remota.

A sensibilidade de alguns sensores dependentes de algumas condições climatéricas, como o PIR com a temperatura, o micro-ondas e ultrassons com chuvas e ventos fortes (vibração), é um dos parâmetros importantes de reconfiguração, uma vez que pode levar a falsas leituras, comprometendo o bom funcionamento do sistema. Qualquer uma das topologias abordadas no capítulo anterior poderá ter capacidade para ser remotamente gerida, desde que tenha incorporado um módulo de comunicação.

Em sistemas com capacidade para controlar o fluxo luminoso, a criação e alteração de perfis de iluminação é também muito importante para um bom e eficiente funcionamento. Não basta ligar e desligar a iluminação, há a necessidade de criar um perfil de iluminação, que quando ocorre movimento, a iluminação fornecida e a forma como é fornecida proporcione a maior comodidade a quem passa, com a melhor eficiência possível. Este tópico será abordado com maior detalhe na próxima secção.

3.3 - Iluminação

Como se pode observar no diagrama da figura 9, o módulo de iluminação é composto pela alimentação (preferencialmente controlável) e pela fonte luminosa. É imprescindível que estes tenham uma grande eficiência, pois serão os responsáveis por quase todo o gasto energético quando comparados com os restantes módulos, embora com a utilização de fontes de tecnologia SSL, como os LED's e a existência de *LED drivers* de grande eficiência, seja possível obter-se um sistema bastante eficiente.

A comodidade e a segurança das pessoas têm também um peso importante. Não se pode simplesmente ligar e desligar instantaneamente a iluminação quando existe movimento, havendo portanto a necessidade da criação de um perfil de iluminação como se pode observar na figura 24.

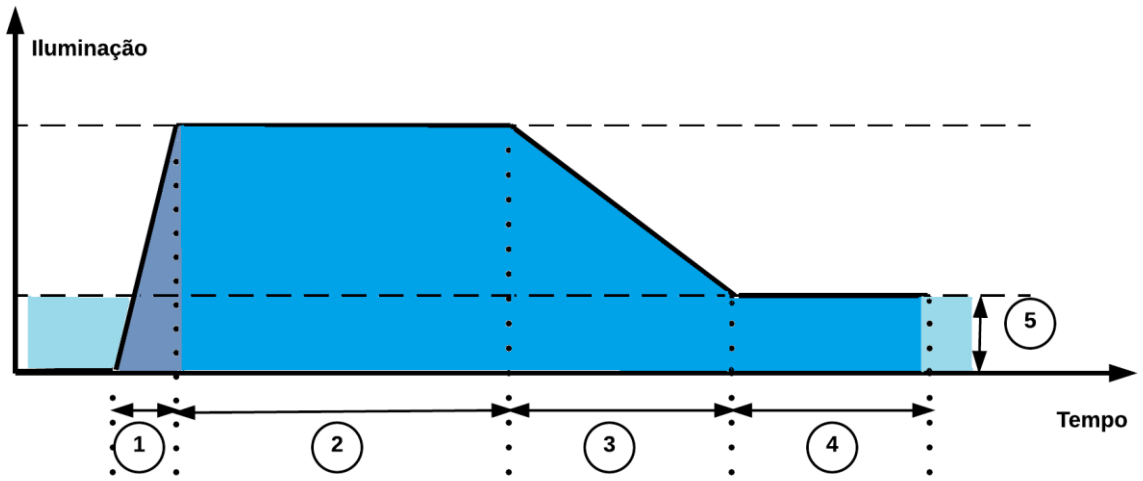


Figura 24 - Exemplo de um perfil de iluminação [10].

Um perfil de iluminação caracteriza o comportamento que a luminária tem ao longo de um determinado período de tempo. Neste tipo de sistemas em particular, esse perfil irá repetir-se sempre que for detetado movimento. Um perfil é composto por várias fases:

1- Instante em que é detetado movimento: a iluminação sobe até ao valor máximo pré-definido de forma instantânea, ou mais suave (com um tempo de *fade-in* associado);
2- Tempo em que a iluminação está na sua intensidade máxima;
3- Tempo em que a iluminação vai diminuindo até ao seu valor mínimo (*fade-out time*);
4- Tempo em que a iluminação está na sua intensidade mínima;
5- Há situações em que não se pretende desligar completamente a iluminação. Desta forma, mesmo na ausência de movimento, o valor mínimo de iluminação prolongar-se-á até nova deteção e o tempo do ponto 4 poder-se-á dizer infinito.

3.4 – Comunicação

Como se viu na secção anterior sobre sensores, há muita informação a ser recolhida e tratada. Dessa forma, é indispensável existir comunicação entre os diversos elementos do sistema.

Sendo um sistema de iluminação pública, é importante haver troca de informação entre as luminárias, seja para partilha dos dados recolhidos pelos sensores (pois em muitos casos, não há a necessidade de todas as luminárias estarem equipadas com os mesmos sensores) ou para controlo de falhas, uma vez que assim, mesmo que determinada luminária não esteja a funcionar corretamente na recolha de dados, pode usar a informação das luminárias vizinhas, até que o problema seja corrigido.

A informação de movimento detetado por uma luminária vizinha, é outro exemplo de informação que pode ser extremamente útil para a unidade de controlo, pois permite que esta se “prepare” para uma provável deteção de movimento, seja atuando sobre alguns sensores ou diretamente sobre a iluminação.

Tal como a comunicação entre os constituintes da luminária, a troca de informação com o exterior desempenha um papel muito importante. Essa informação pode ser para monitorização das diversas variáveis do sistema, isto é, informação que provém do sistema para fora, ou de configuração (de fora, para o sistema), permitindo que alguns parâmetros do sistema sejam alterados de forma remota.

Existem diversos protocolos que se poderiam utilizar para a troca de toda esta informação, sejam estes com ou sem fios.

Será apresentado um estudo sobre alguns protocolos, mas primeiramente é importante falar sobre as topologias de rede que podemos utilizar.

3.4.1 - Topologias de rede

Antes de se avançar com o estudo de alguns protocolos de comunicação, é importante assimilar alguns conceitos e topologias relativos a estes, de forma a facilitar a sua compreensão. Dessa forma, segue-se uma breve exposição das principais topologias de rede, utilizadas nos diversos protocolos de comunicação.

Ponto-a-ponto

É a mais simples de todas as topologias, onde existem apenas dois nós que comunicam diretamente entre si.

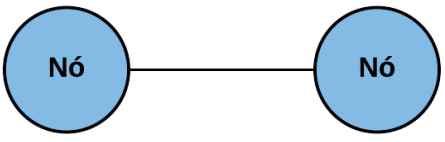


Figura 25 - Topologia de comunicação ponto-a-ponto.

Barramento

Nesta topologia em barramento ou *bus*, todos os nós estão ligados através de um barramento central (*backbone*) que é partilhado para comunicarem entre si. O facto de qualquer quebra no barramento inutilizar toda a rede é uma das principais limitações desta topologia.

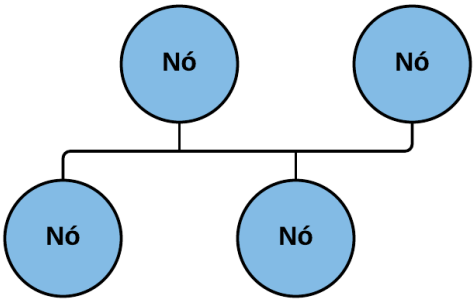


Figura 26 - Topologia de comunicação em barramento.

Estrela

Todos os nós integrantes da rede estão ligados a um mesmo nó (nó central ou *hub*), e toda a informação passa obrigatoriamente por ele. De forma idêntica à topologia em barramento, aqui existe uma dependência total do nó central pois uma falha neste nó e toda a rede fica inutilizada.

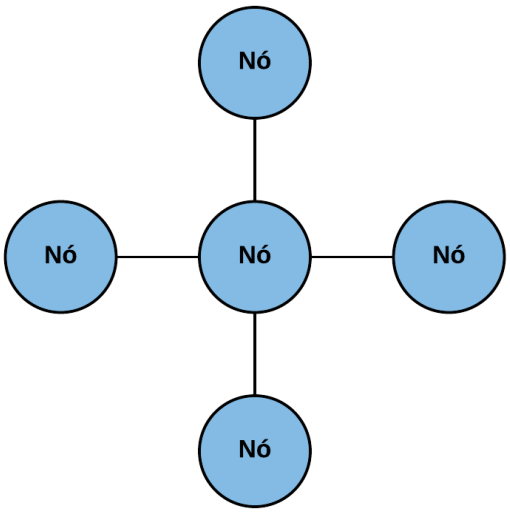


Figura 27 - Topologia de comunicação em estrela.

Anel

Na topologia em anel, os nós estão interligados formando um círculo, e toda a informação vai passando pelos nós até ao nó destino. Mesmo havendo uma quebra numa das ligações, há a possibilidade da informação continuar a fluir, caso seja uma topologia em anel bidirecional.

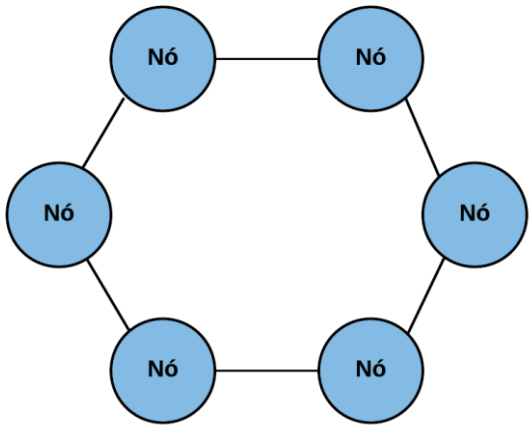


Figura 28 - Topologia de comunicação em anel.

Malha

Podem-se encontrar dois tipos de topologias em malha ou *mesh*: a completa e a parcial. Na topologia de malha completa, cada nó, está ligado diretamente aos restantes, oferecendo uma enorme robustez à rede, mas a um custo muitas vezes incomportável. O número de ligações necessárias numa rede deste tipo, pode ser calculado em relação ao número de nós (n) pela expressão: $n(n-1)/2$. Pode-se observar um exemplo desta topologia na figura 29 a).

Na topologia em malha parcial, o número de ligações é variável, havendo um compromisso entre robustez e custo na determinação do número de ligações a efetuar. Pode-se observar um exemplo desta topologia na figura 29 b).

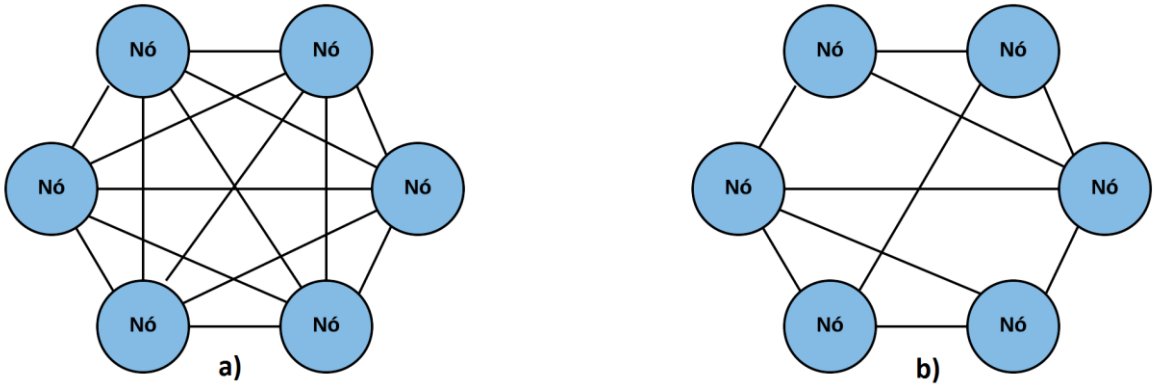


Figura 29 - Topologia de comunicação em malha completa [a)] e malha parcial [b)].

Árvore

Esta topologia é constituída por um nó denominado de nó raiz ou *root,* que pode comunicar com vários nós e cada um destes nós, por sua vez, podem comunicar com vários outros nós e assim sucessivamente, criando a forma de uma árvore.

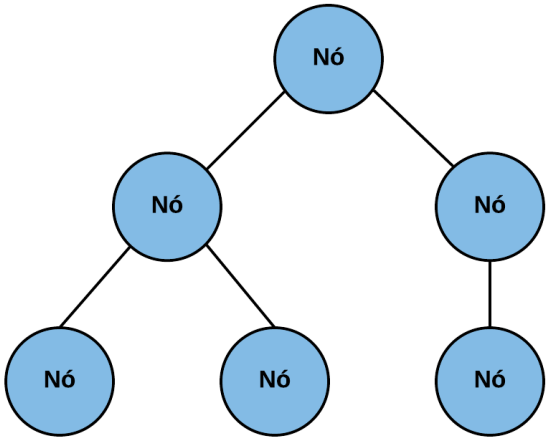


Figura 30 - Topologia de comunicação em árvore.

3.4.2 - Comunicação com fios

Protocolo CAN (Controller Area Network)

Desenvolvido pela Robert Bosch em meados da década de 80, este protocolo visava a redução da cablagem necessária na indústria automóvel. Com o aumento dos dispositivos eletrónicos integrantes dos automóveis, a quantidade de cablagem necessária para comunicação era elevada. Com este protocolo tornou-se possível que todos os dispositivos partilhassem o mesmo barramento de dados, ou seja, a mesma cablagem.

Este protocolo de comunicação série, definido na norma ISO-11898 caracteriza-se principalmente pela sua robustez. Possui acesso múltiplo ao barramento sensível à portadora, com deteção de colisões e arbitragem na prioridade da mensagem (CSMA/CD+AMP). CSMA (*Carrier-Sense, Multiple Acess*) significa que cada nó irá esperar um determinado tempo de inatividade antes de tentar enviar uma mensagem. CD+AMP (*Collision Detection + Arbitration on Message Priority*) indica que as colisões são resolvidas com arbitragem *bit-a-bit* onde a mensagem com o identificador mais baixo tem a prioridade mais alta.

Cada nó do barramento CAN consegue detetar erros nas mensagens forçando-as a serem destruídas e retransmitidas. Antes de qualquer mensagem poder ser

processada, todos os nós têm que a validar, sendo que essa validação só é enviada depois de várias verificações de erros, bastando apenas um nó encontrar um erro para inviabilizar a mensagem.

Este tratamento de erros é possível, porque cada mensagem transmitida não é endereçada a nenhum nó específico mas sim a todos (*broadcast*), contendo na mensagem um valor numérico de controlo de prioridade no barramento que pode ser utilizado como identificador, cabendo depois a cada nó decidir se vai atuar ou não sobre a mensagem. O protocolo permite também que se retire ou adicione um nó sem necessidade de atualizar os nós já existentes.

Tal como já referido, o CAN é um protocolo série que utiliza duas linhas de comunicação diferenciais (CANH e CANL) complementares como se pode observar na figura 31.

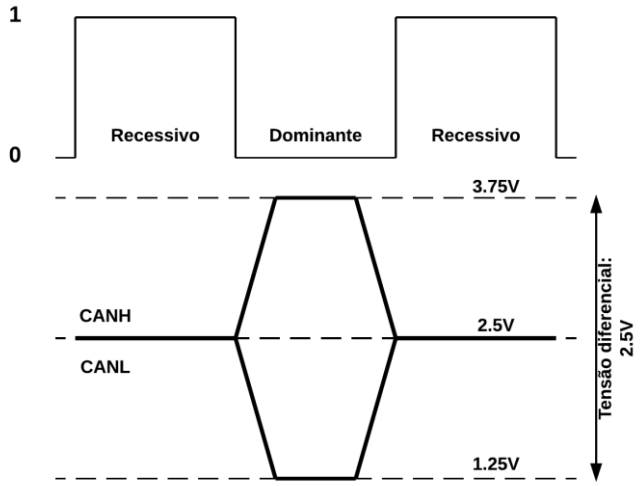


Figura 31 - Demonstração dos sinais diferenciais presentes num barramento de CAN.

Geralmente os *transceivers*[8] de CAN encontrados no mercado, operam a $3.3V$ ou $5V$, sendo que nas condições da norma ISO11898, para taxas elevadas de transmissão de dados, é recomendado um máximo de 30 nós [11] a partilhar o mesmo barramento (com terminação de linha de 120Ω) que pode chegar aos $1000\,m$ de comprimento e uma taxa máxima de $1\,Mbit/s$, que pode decair para $50\,Kbit/s$ a distâncias perto do máximo permitido.

[8] *Transceiver* ou transcetor (em português), é um dispositivo que acumula as funções de transmissor e recetor.

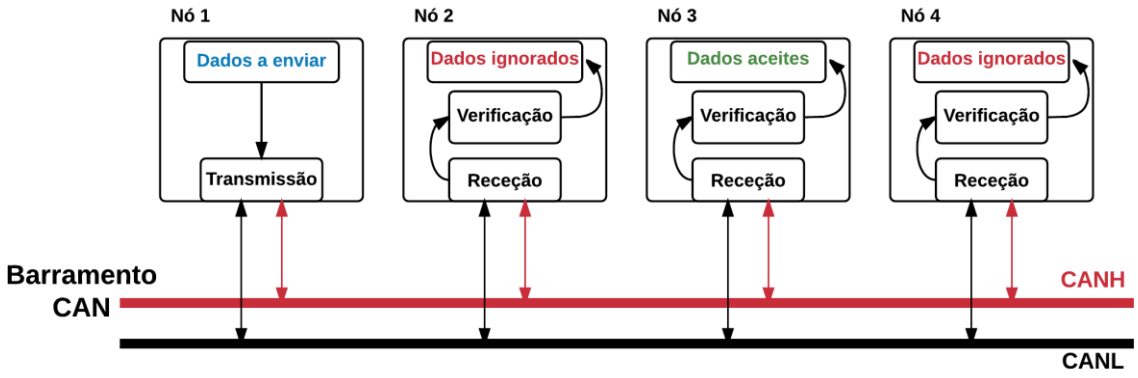


Figura 32 - Exemplo de transmissão de dados num barramento CAN.

Na figura acima é possível observar um exemplo de transmissão de dados do nó 1 para o nó 3. O nó 1 envia os dados pelo barramento, que são recebidos pelos restantes nós. Após verificação, o nó 3 confere que os dados se destinam a ele e aceita-os. Os restantes nós, descartam a informação recebida.

Comunicação pela rede elétrica (PLC)

A tecnologia PLC (do inglês, *Power Line Communication*) é tal como o nome indica, uma forma de comunicação que utiliza a infraestrutura da rede elétrica como via de comunicação. Por volta de 1922, o primeiro sistema de transmissão de informação através da rede começou a operar em linhas de alta-tensão utilizando a gama de frequências entre os $15 - 500\ KHz$ para telemetria. Mais tarde na década de 70, a *Tokyo Electric Power Co* reportou que teria conseguido comunicação bidirecional através da rede elétrica [12]. Apesar da vantagem do uso da rede elétrica como infraestrutura de comunicação, é preciso entender que estas não foram criadas com o objetivo de transmitir dados, sendo na verdade, um ambiente extremamente difícil para esse propósito devido a ruídos e interferências, e onde os valores de impedância não são constantes, criando atenuação e reflexões na linha que inviabilizam muitas vezes, os sinais de dados que nela fluem. Posto isto, existe a necessidade de sistemas complementares que tentem contornar todos estes problemas dando por um lado robustez ao protocolo, mas por outro encarecendo-o.

RS-485

É um protocolo de comunicação série definido pela norma EIA/TIA (*Electronic Industries Association/Telecommunications Industries Association*). Embora continue a ser comummente denominado por RS-485, onde a sigla RS provém de *Recommended Standard*, o nome oficial é TIA-485. Pertencente à família de um dos mais antigos protocolos de comunicação série (RS-232), este utiliza duas linhas de comunicação diferencial com tensões máximas compreendidas entre os $-7V$ e os $+12V$. Permite a criação de uma rede de comunicação com um máximo de 32 transmissores e 32 recetores e uma distância máxima de cablagem de $1200m$ com terminação de linha de

100Ω. Suporta transferências de dados a $10\,Mbit/s$ sendo que esta pode cair para menos de $100\,Kbit/s$ para distâncias perto do máximo permitido.

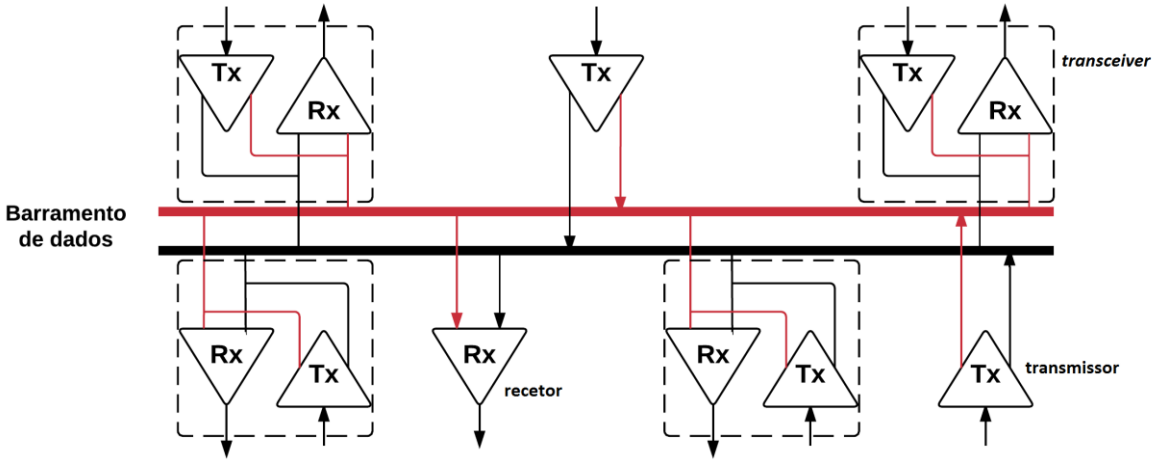


Figura 33 - Demonstração de um barramento de RS-485.

3.4.3 – Comunicação sem fios (*Wireless*)

As tecnologias de comunicação sem fios, são de longe aquelas com maior crescimento na área das comunicações. Estas permitem a transferência de informação sem a utilização de qualquer condutor elétrico, mas sim através da propagação de ondas pelo ar. A informação pode viajar desde alguns metros, por infravermelhos por exemplo, como quando utilizamos o comando da televisão, até centenas de quilómetros, como nas comunicações rádio.

WiFi

Em 1970 a Universidade do Havai desenvolveu a primeira rede de comunicação sem fios (ALOHANET), permitindo a transferência de dados entre ilhas desse país. Duas décadas passaram e em 1990 o IEEE (*Institute of Electrical and Electronics Engineers*) iniciou a discussão para a normalização das tecnologias de rede local sem fios (WLAN, do inglês *Wireless Local Area Network*). Em 1997, após 7 anos de pesquisa e desenvolvimento, o padrão IEEE 802.11 foi aprovado, sendo geralmente denominado por WiFi.

Provavelmente a tecnologia mais ubíqua de entre todas as tecnologias sem fios, essencialmente devido à sua relação e associação com a internet, utiliza geralmente a banda ISM[9] nos $2.4\,GHz$, permitindo que dispositivos eletrónicos (em topologia ponto-

[9] Banda ISM (do inglês, *Industrial, Scientific and Medical radio bands*), são porções do espectro de rádio frequência (RF) não licenciadas e livres para serem utilizadas. Existem várias gamas de frequências que podem ser utilizadas livremente sejam localmente ou a nível mundial, sendo a mais conhecida e utilizada os $2.4\,GHz$ [13].

multiponto) troquem informação através de redes WLAN, com taxas de transferência muito elevadas. Estes protocolos implicam um elevado consumo energético, que o pode impossibilitar em determinadas aplicações, tendo no entanto uma grande vantagem quando existe necessidade de acesso à internet.

Bluetooth/BLE

Este protocolo foi idealizado e iniciado pela companhia de telecomunicações sueca Ericsson em 1994 como uma alternativa sem fios ao protocolo RS-232. A ideia de interligar vários dispositivos sem o uso de fios, rapidamente chegou a outras empresas. A primeira normalização foi a IEEE 802.15.1, mas a partir da criação de um grupo, o *Bluetooth SIG (Bluetooth Special Interest Group)*, formado inicialmente pela Ericsson, Nokia, Intel, Toshiba e a IBM, todos os assuntos relacionados com normalizações ou licenciamentos de tecnologias *Bluetooth* são da sua responsabilidade. Como curiosidade, o nome surge relacionado com o rei Harold I da Dinamarca, também designado como Harold Dente-Azul (Bluetooth), que ficou conhecido de entre outros feitos, pela unificação da Dinamarca, algo que este grupo ambicionava fazer, metaforicamente, com os dispositivos móveis através desta tecnologia.

Desenvolvida para comunicações ponto-a-ponto de curtas distâncias, apresenta uma grande robustez e segurança, aliado a um baixo consumo, especialmente depois do aparecimento do *Bluetooth Low Energy* (BLE) ou *Bluetooth Smart*.

Zigbee

Criado pela Zigbee Alliance por volta de 1998, como forma de colmatar certas lacunas que o WiFi e o Bluetooth apresentavam em determinadas aplicações, nomeadamente a nível de consumo energético, complexidade e custo. Aplicações que tinham no baixo consumo, o seu principal objetivo, mesmo tendo que abdicar de quantidade e velocidade de informação a ser transmitida. Em 2003, a normalização IEEE 802.15.4 foi concluída e nos anos seguintes o número de membros foi crescendo, possuindo hoje em dia a colaboração de mais de 200 entidades [14].

Tal como o WiFi e o Bluetooth, opera na gama dos $2.4\,GHz$, e foi criado essencialmente para redes de sensores sem fios e automação. É utilizado geralmente em topologias de rede do tipo malha, mas também em estrela e árvore.

É um protocolo que apresenta baixos custos, com baixo consumo e fácil implementação, aliados a uma grande robustez e segurança. Em contrapartida, possuí baixas taxas de transmissão e alcance relativamente curto. Essa lacuna pode ser contornada com a topologia de rede utilizada, uma vez que os dispositivos Zigbee podem passar a informação entre os nós de forma a alcançarem nós que se encontrem em pontos mais distantes na rede.

Devido à possibilidade de ter um elevado número de nós na rede (até 65.000) quando comparado com outras tecnologias, e podendo todos eles comunicarem entre si,

fez lembrar as abelhas a trabalharem numa colmeia, voando em *zig-zag* e trocando informação entre si, dando origem ao nome Zigbee.

Na figura 34, pode-se observar um exemplo de uma rede Zigbee e dos seus constituintes.

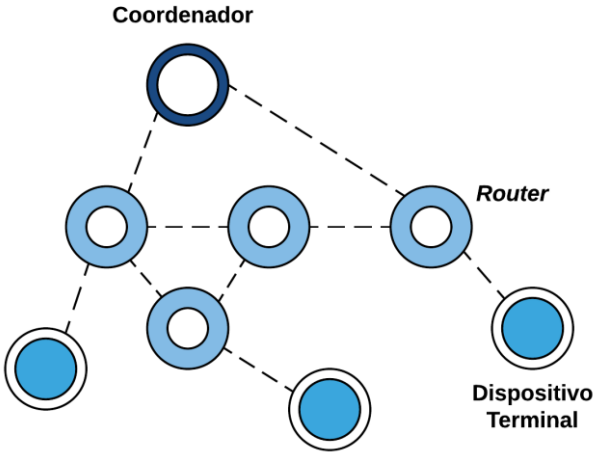


Figura 34 - Exemplo de uma rede Zigbee [15].

Um sistema Zigbee, é constituído por três tipos de dispositivos: o coordenador, os transportadores de informação ou intermediários (*Routers*) e os dispositivos terminais, que são aqueles a que se pretende aceder. O Coordenador funciona como nó raiz (*root*) da rede. Tal como o nome indica, é o coordenador da rede, e é responsável pela receção e transmissão de dados. Os *Routers* têm como função, levar a informação para o Coordenador e para os dispositivos terminais. Estes podem ser dispositivos como sensores ou atuadores.

Proprietary ISM

São protocolos que ao contrário dos anteriores aqui falados, não estão sujeitos a qualquer normalização ou *standard*, e como tal, são específicos da entidade que os concebeu, ou seja, do seu proprietário. Desta forma, se quisermos uma rede funcional, teremos obrigatoriamente que utilizar *transceivers* da mesma empresa/proprietário, uma vez que não há garantias, que sejam compatíveis com outros dispositivos. Apesar da evidente desvantagem, algumas marcas fizeram avanços de tal forma notórios, essencialmente em dispositivos *ultra low power*, que a falta de interoperabilidade dos mesmos com outros dispositivos, deixou de ser um problema e hoje em dia é vista como uma boa solução em diversas aplicações.

3.4.4 - Conclusões

Neste capítulo, observámos a diversidade de informação possível de ser recolhida através do uso de sensores. Existe uma enorme variedade de sensores com o mesmo propósito, mas com características diferentes e a escolha daquele que melhor se adequa é fundamental.

Além da recolha de toda essa informação, é importante dotar o sistema de capacidade de comunicação, seja para enviar todos esses dados recolhidos, ou para receber informação do exterior. Portanto, a escolha do protocolo de comunicação que melhor se adequa ao sistema é igualmente importante. As opções são muitas, desde protocolos com fios (*wired*) aos sem-fios (*wireless*).

Os protocolos com fios permitem taxas de transferência de dados mais elevadas e maior segurança. No entanto, a necessidade de cablagem traz custos e limitações. Essas limitações são contornadas pelos protocolos *wireless*, que oferecem grande mobilidade e permitem um maior alcance, ainda que sejam mais lentos e menos seguros. Os avanços nestas tecnologias, têm permitido taxas de transferência de dados cada vez mais elevadas e seguras, debelando assim algumas das desvantagens destes protocolos.

No caso deste sistema em particular, as vantagens da utilização de um protocolo *wireless* são muito interessantes, essencialmente a sua mobilidade e a ausência de cablagem adicional para comunicação. O PLC seria também uma boa opção, contudo a sua implementação acarretaria maiores custos e teria um grau de dificuldade mais elevado, embora dentro dos protocolos com fios, seja o principal a ter em conta para um sistema de IP.

	WiFi	**Bluetooth (Smart/BLE)**	**Zigbee**	**Proprietary ISM**
Normalização	Sim	Sim	Sim	Não
Consumo Energético	Elevado	Baixo	Médio	Variável
Taxa de transmissão	$+100\ Mbps$	$1\ Mbps$	$250\ Kbps$	Variável
Alcance	$100m$ – alguns Km	$50 - 150\ m$	$10 - 100\ m$	Variável
Banda de frequência	$2.4\ GHz$	$2.4\ GHz$	$2.4\ GHz$	Variável
Número de dispositivos na rede	Elevado	Reduzido	Elevado	Variável

Tabela 1 - Resumo dos protocolos *wireless*.

Na tabela 1 pode-se observar a comparação de algumas características dos protocolos *wireless* abordados. O WiFi pelo consumo, e o Bluetooth pelas limitações na rede principalmente, partem atrás do Zigbee, que além de um consumo reduzido foi criado para redes do tipo malha, possuindo capacidade para uma rede com milhares de dispositivos ligados.

Capítulo 4

4 - Sistema Desenvolvido

Como se viu ao longo desta dissertação, um sistema de IP é composto por vários módulos (ver figura 9). Na primeira parte deste capítulo será abordado o módulo de controlo da luminária e mostradas as diversas fases do seu desenvolvimento.

Na segunda parte do capítulo, será apresentado o módulo de gestão remota concebido para gerir (monitorizar e configurar) o módulo anteriormente mencionado.

4.1 – Módulo da Luminária

Na presente secção, serão indicadas e demonstradas as várias etapas da criação do módulo da luminária, desde o projeto em *breadboard* até á produção do protótipo final desenvolvido em placa de circuito impresso (PCB) para aplicação na luminária a instalar no piloto do projeto LITES.

4.1.1 – Constituintes do módulo

Será feita de seguida, uma apresentação dos vários elementos que constituem este módulo.

Unidade de controlo

A unidade de controlo escolhida foi o Arduino Nano. É uma placa de desenvolvimento de pequenas dimensões e fácil aplicação em placa branca devido à forma dos seus pinos, como se pode observar na figura 35.

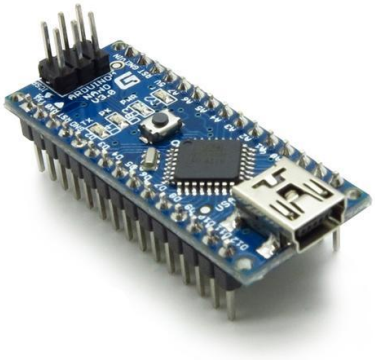


Figura 35 - Placa de desenvolvimento Arduino Nano.

Além da sua pequena dimensão e compatibilidade com placa branca, outra das vantagens é a sua fácil utilização, bastando um cabo mini-USB e a instalação do *software* do ambiente de desenvolvimento ou IDE (do inglês, *Integrated Development Environment*) para trabalhar com ele, não havendo necessidade de hardware adicional para programação, devido ao *bootloader*[10] que traz gravado na memória *flash*. Esta placa contém um microcontrolador *RISC*[11] da família $Atmel\ AVR$ de 8 bits, o $ATmega328P$. Na tabela 2 é possível observar as principais características da placa.

Microcontrolador	$ATmega328P$
Tensão de operação	$5V$
Tensão de alimentação recomendada	$7 - 12V$
Entradas/Saídas digitais	$14\ (das\ quais\ 6\ permitem\ PWM)$
Entradas analógicas	8
Corrente DC por pino	$40\ mA$
Memória Flash	$32\ KB$
SRAM	$2\ KB$
EEPROM	$1\ KB$
Frequência de operação	$16\ MHz$
Comunicação	$UART, SPI, I^2C$
Dimensões	$18\ \times\ 45\ mm$

Tabela 2 - Características da placa de desenvolvimento Arduino Nano [16].

Comunicação

Para efeitos de comunicação, a escolha recaiu sobre um protocolo *wireless*. O módulo escolhido utiliza o chip $nRF24L01+$ da $Nordic$, que é um *transceiver ultra low power* de radiofrequência (RF) que opera nas bandas ISM, mais concretamente nos $2.4\ GHz$, com um amplificador de potência. Permite a transferência de pacotes de 1 a 32 *bytes* a taxas de $250\ Kbps$, $1\ Mbps$ e $2\ Mbps$. Pode ser utilizado e configurado com um microcontrolador, através do protocolo de comunicação SPI e apresenta consumos de aproximadamente $11.3\ mA$ a transmitir a $0\ dBm$ e $13.5\ mA$ em modo RX, a receber a $2\ Mbps$ [17]. Na tabela 3 pode-se ver algumas características do módulo utilizado.

[10]Um *bootloader* no contexto dos microcontroladores é um programa que corre nestes dispositivos, concebendo-lhes capacidades adicionais, nomeadamente de poderem ser programados sem recurso aos programadores específicos para o efeito).

[11]Acrónimo para *Reduced Instruction Set Computer* é uma arquitetura de processadores com um número reduzido de instruções simples que têm aproximadamente o mesmo tempo de execução.

Figura 36 - Módulo de comunicação nRF24L01+ da Nordic.

Frequência de operação	$2.4\ GHz$
Número de canais	126
Taxas de transmissão possíveis	$250\ Kbps, 1\ Mbps\ e\ 2\ Mbps$
Potências de transmissão possíveis	$0, -6, -12, -18\ dBm$
Dimensões do pacote	$1\ a\ 32\ bytes$
Tensão de alimentação	$1.9\ -\ 3.6V$
Consumo em *Standby*	$26\mu A$
Consumo em *Power Down mode*	$900nA$
Consumo em modo TX	$11.3\ mA\ @\ 0\ dBm$
Consumo em modo RX	$13.5\ mA\ @\ 2Mbps$
Interface de comunicação	SPI

Tabela 3 - Características do módulo nRF24L01+ [17].

Iluminação

O bloco de iluminação do sistema é constituído por uma fonte de alimentação controlável, responsável por fornecer energia à fonte luminosa. De seguida será feita uma apresentação dos constituintes deste bloco.

- ## Alimentação

Para fonte de alimentação, optou-se por uma fonte comutada de $40W$ da $Mean\ Well,\ NPF-40D-54$ com capacidade de controlo por PWM. Além de permitir o controlo, a elevada eficiência, as várias proteções e robustez fizeram a escolha recair por este *driver* comercial, uma vez que este terá que funcionar em cenário real com

condições adversas. Algumas das suas especificações podem ser observadas na tabela 4.

Potência	$41.04W$
Tensão de Saída (DC)	$54V$
Tensão de Entrada (AC)	$115VAC, 230VAC\ e\ 277VAC$
Eficiência	90%
Correção do fator de potência (tip.)	> 0.95
Proteções	Curto-circuito, sobre-corrente, sobre-tensão, sobre-temperatura e IP67
Dimensões	$150 \times 53 \times 35\ (mm)$

Tabela 4 - Características da fonte NPF-40D-54 da Mean Well [18].

Figura 37 - Fonte comutada NPF-40D-54 da Mean Well.

- ## Dispositivo de iluminação

Pelas suas vantagens em relação a outras tecnologias de iluminação e por ser um requisito do projeto LITES, utilizou-se um dispositivo de iluminação de tecnologia LED. Ao contrário do anterior, aqui decidiu-se manter a solução já implementada com a utilização de duas fitas de LED's da $Tridonic$, $STARK - LLE\ 24 - 280 - 1250 - 840 - CLA$. Na tabela 5 é possível observar algumas das suas características. Em conjunto, é utilizado também um difusor responsável por distribuir a luz de forma homogénea.

Figura 38 - Módulo LED STARK-LLE 24-280-1250-840-CLA da Tridonic.

Número de LEDs	22
IRC	> 80
Tempo de vida	$até\ 50.000\ horas$
Temperatura de cor	$4000K$
Fluxo Luminoso (típico)	$1360\ lm$
Potência	$11.8W$

Tabela 5 - Características do módulo STARK-LLE24-280-1250-840-CLA [19].

Sensores

- ## Temperatura e Humidade

Com a diversidade de sensores existentes, optou-se por uma solução integrada de baixo custo e boa estabilidade, com capacidade para medir temperatura e humidade relativa. A medição de temperatura é feita por um termístor do tipo NTC, enquanto o sensor de humidade é do tipo capacitivo. Além destes dois sensores, o dispositivo contém um microcontrolador de 8 bits integrado, que permite uma saída digital, que utiliza um protocolo $1-wire$ para troca de informação com o exterior. O dispositivo utilizado é o $AM2302$ (também conhecido por $DHT22$) da $AOSONG$, que se pode observar na figura 39, tendo as suas principais características evidenciadas na tabela 6.

Figura 39 - Sensor integrado de temperatura e humidade DHT22.

	Temperatura	Humidade
Tensão de alimentação	$3.3 - 5.5\ V$	
Consumo (Medição)	$500\mu A$	
Consumo (*Standby*)	$15\ \mu A$	
Gama de medição	$-40\ a\ 80℃$	$0\ a\ 99.9\%$
Resolução	$0.1℃$	0.1%
Precisão	$\pm 0.5℃$	$\pm 2\%$
Comunicação	$1-wire$	

Tabela 6 - Característica do sensor DHT22 [20].

- ## Corrente

Para a medição do consumo, optou-se por um transformador de corrente de forma a garantir isolamento galvânico e a não necessidade de alteração do circuito projetado.

O campo magnético gerado pela passagem de corrente elétrica num fio condutor, ao atravessar o sensor, dá origem a uma corrente proporcional à medida (numa relação de 1000:1). Essa corrente gerada passa posteriormente por uma resistência de 200Ω, dando origem a uma diferença de potencial compreendida entre 0 e $1V$. O sensor utilizado é o $TA12-100$, indicado para medir corrente alternada até $5A$ e que pode ser observado na figura 40.

Figura 40 - Transformador de corrente TA12-100.

Transformação/Relação	$1000:1$
Gama de medição	$0 - 5\ A$
Corrente de amostragem	$0 - 5\ mA$
Tensão de amostragem	$0 - 1V$

Tabela 7 - Características do TA12-100 [21].

- ## Vibração

Para deteção de vibração na luminária, optou-se por um acelerómetro digital de 3 eixos. O sensor escolhido foi o $ADXL345$ da $Analog\ Devices$, com resolução até 13 bits e suporte de comunicação por I^2C e SPI. Apresenta um baixo consumo de cerca de $23\mu A$ quando em medição e um sistema de gestão de memória do tipo FIFO com 32 níveis que permite armazenar os dados recolhidos, libertando assim o processamento do microcontrolador e consequentemente, baixando o seu consumo [29]. Na figura 41 é possível observar o módulo utilizado, que para além do sensor, já possui a eletrónica adicional requerida por este. O módulo contém também um regulador linear de $3.3V$ que permite a alimentação a $5V$ e resistências de *pull-up* para as linhas de I^2C.

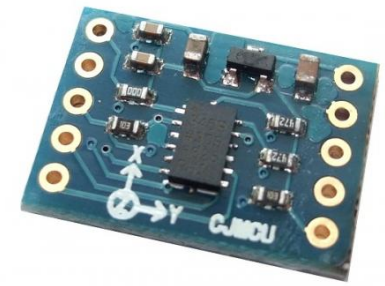


Figura 41 - Placa com acelerómetro ADXL345.

Tensão de operação	$3.3V$
Consumo (Operação)	$23\mu A$
Consumo (*Standby*)	$1\mu A$
Resolução	$10\ -13\ bits$
Comunicação	$I^2C\ e\ SPI$
Sensibilidade	$\pm 2g\ \pm 4g\ \pm 8g\ \pm 16g$

Tabela 8 - Características do ADXL345 [22].

- **Pressão atmosférica**

Para a medição desta variável, escolheu-se um sensor digital da $Bosch$, o $BMP180$. É um sensor de baixo consumo ($5\mu A$, quando a trabalhar a $1\ amostra/s$), e gama de medição entre os 300 e os $1100\ hPa$. Possui interface de comunicação por I^2C e sensor de temperatura integrado [23]. Tal como na escolha do acelerómetro, optou-se por um módulo com a eletrónica adicional já integrada que pode ser observada na figura 42.

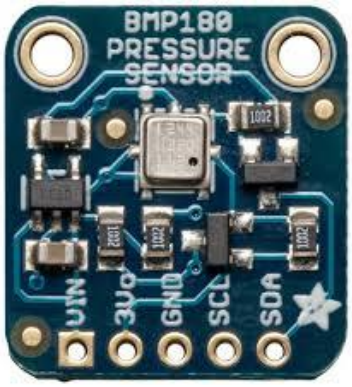


Figura 42 - Placa com o BMP180 integrado.

	Pressão	**Temperatura**
Tensão de operação	3.3V	
Consumo (Operação)	$5\ \mu A$ @ $1\ amostra/s$	
Consumo (*Standby*)(típ.)	$0.1\ \mu A$	
Gama de medição	$300 - 1100\ hPa$	$0 - 65$ ℃
Precisão	$-1\ \pm 1\ hPa$	± 1 ℃

Tabela 9 - Características do sensor BMP180 [23].

- **Sensor de movimento passivo**

Para a deteção de movimento de forma passiva, escolheu-se um sensor optoeletrónico de infravermelhos, um sensor PIR. O sensor escolhido foi o $LHi\ 968$ da $excellitas$, que possui dois elementos sensoriais (ver secção 3.1.5) e pode ser observado na figura 43. Na tabela 10 é possível ver algumas das principais características do sensor.

Figura 43 - Sensor optoeletrónico de infravermelhos passivo (PIR).

Tensão de Operação	$2 - 12V$
Elementos sensoriais	2
Ruído (típ.)	$20\ \mu V_{PP}$ @ $0.4 - 10Hz$ e 20 ℃
FoV (montagem horizontal)	$100°$

Tabela 10 - Características do sensor PIR LHi 968.

- ## Sensor de movimento ativo

Para sensor de movimento ativo, a escolha recaiu sobre um sensor de micro-ondas. Pelo facto de os sensores de ultrassons (que também poderiam ser uma opção a ter em conta) serem mais suscetíveis a condições atmosféricas e o local do piloto estar sujeito a algumas condições adversas, a solução com micro-ondas foi escolhida. O sensor utilizado é o $MDU1750$ da $microwave\ solutions$, que opera na $banda\ X$ ($8 - 12\ GHz$) das micro-ondas e que faz uso do efeito de *Doppler* para a deteção de movimento, apresentando um baixo custo e consumo. O sensor pode ser visto na figura 44 e na tabela 11 é possível observar algumas características.

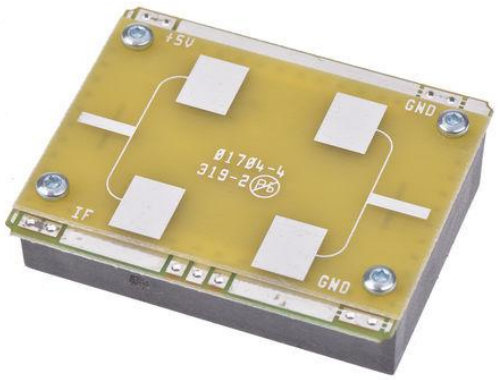


Figura 44 - Sensor de micro-ondas MDU1750.

Tensão de alimentação	$5 \pm 0.25V$
Banda de Operação	$Banda - X\ (8 - 12GHz)$
Consumo em modo contínuo (típ.)	$40\ mA$

Tabela 11 - Características do sensor de micro-ondas MDU1750.

4.1.2 – Protótipo experimental

Sendo o sistema constituído por vários módulos, há a necessidade de desenvolvimento e testes de cada um deles de forma independentemente.

Numa primeira fase, optou-se pelo desenvolvimento do sensor híbrido composto pelo sensor PIR e pelo sensor de micro-ondas apresentados na secção anterior. De seguida, vai-se fazer uma apresentação do desenvolvimento efetuado.

- **Sensor híbrido**

  - **Sensor de micro-ondas**

O sinal de saída do sensor é muito pequeno para ser tratado pelo microcontrolador. Dessa forma há a necessidade acondicionar o sinal para que este chegue nas condições adequadas para ser processado. O circuito em questão pode ser observado na figura 45.

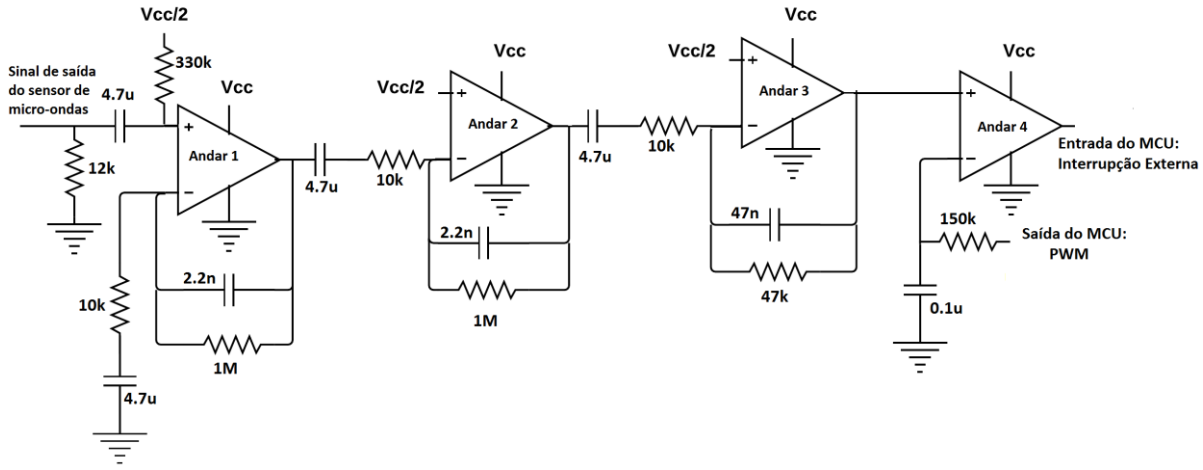


Figura 45 - Circuito de acondicionamento de sinal para o sensor de micro-ondas.

Como se pode observar na figura acima, o circuito é composto por quatro andares utilizando amplificadores operacionais (OpAmp). Iniciando a análise do circuito da figura 45, o primeiro andar é responsável por fazer uma pré-amplificação do sinal e é composto por um filtro passa-banda com frequências de corte de $3.38\ Hz$ e $72.34\ Hz$ e amplificação em configuração não-inversora.

Desta forma, apenas as frequências contidas entre as calculadas acima serão amplificadas, sendo as restantes anuladas e retirando-se assim a componente DC do sinal, bem como ruído proveniente das altas frequências. Como se trata de uma configuração não-inversora, o ganho deste andar é $1 + \frac{1M}{10K} = 101$.

Seguem-se dois andares de amplificação e filtragem compostos por um filtro passa-banda (com as mesmas frequências de corte do primeiro andar), e amplificação numa configuração inversora com ganho de $-\frac{1M}{10K} = -100$ no segundo andar e $-\frac{47K}{10K} = -4.7$ no terceiro. A tensão de modo comum na entrada do OpAmp é definida por $Vcc/2$, que permite uma melhor amplificação do sinal AC.

O ganho total do circuito é obtido pela multiplicação do ganho obtido em cada andar, resultando em $101 \times (-100) \times (-4.7) = 47\ 470$.

Finalmente no último estágio, tem-se um OpAmp a funcionar como comparador. Este irá saturar positivamente, isto é, terá na saída a sua tensão de alimentação positiva sempre que o sinal na entrada positiva (+) for superior ao sinal presente na entrada negativa (-). Caso contrário, diz-se que o OpAmp está saturado negativamente, apresentando na saída a sua tensão de alimentação negativa.

De forma a se obter algum controlo sobre o circuito, o sinal presente na entrada (-) provém do microcontrolador. É um sinal modulado por largura de pulso (PWM), que ao passar pelo filtro RC passa-baixo, resulta num sinal DC que servirá de referência para o comparador, e que poderá ser ajustado (com a variação do $duty - cycle$) entre $0 - 5V$ (Vcc), permitindo assim aumentar ou diminuir o nível de sensibilidade de deteção do sensor.

Verifica-se portanto, que o sinal resultante só terá duas tensões ou níveis possíveis, 0 ou $5V$. Para tratamento deste tipo de sinal (digital), configurou-se o microcontrolador para gerar uma interrupção sempre que for detetada uma transição descendente ($5 \rightarrow 0V$), isto é, sempre que for detetado um pulso no pino correspondente, dando assim informação ao microcontrolador sobre a deteção ou não de movimento.

- **Sensor PIR**

Tal como no caso apresentado anteriormente, o pequeno sinal na saída do sensor requer acondicionamento para posterior tratamento pelo microcontrolador. Na figura 46 é possível observar o circuito projetado para o efeito.

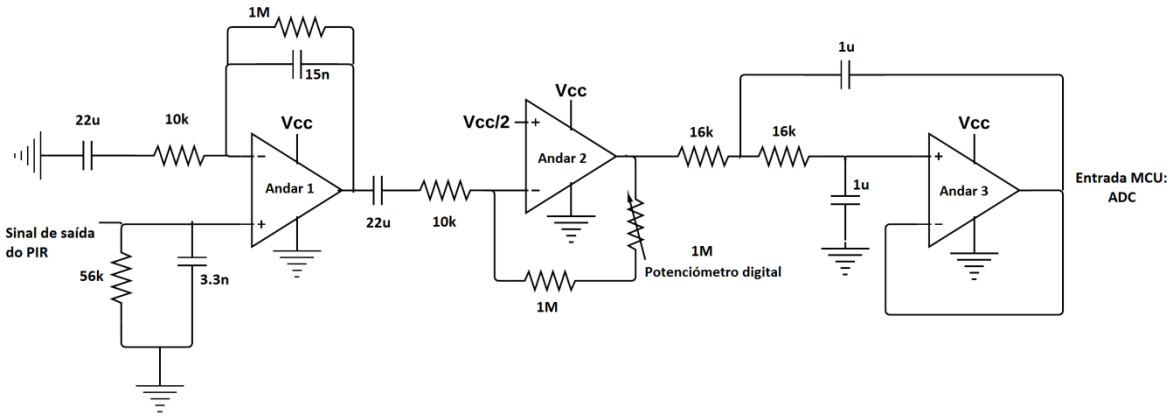


Figura 46 - Circuito de acondicionamento de sinal para o sensor PIR.

O primeiro andar deste circuito é em tudo idêntico ao visto no circuito anterior, possuindo um filtro passa-banda com frequências de corte de $0.72\,Hz$ e $10.61\,Hz$ inferiores e superiores respetivamente e amplificação com um ganho de $1 + \frac{1M}{10K} = 101$.

O andar 2 é responsável pela alteração do ganho do circuito. É utilizado um potenciómetro digital de $1M\Omega$ na posição de *feedback* da configuração. O componente escolhido foi o $AD5241$ da $Analog\ Devices$ [24] com 256 posições possíveis. A componente DC do sinal de saída do primeiro estágio é anulada pelo filtro passa-alto. A configuração é inversora e o ganho pode variar entre -100 e -200. Consequentemente, o ganho total do circuito pode variar entre -10100 e -20200 dependendo do valor do potenciómetro.

Por fim no andar 3, temos um filtro passa-baixo de segunda ordem e ganho unitário com topologia $Sallen - Key$ e frequência de corte $fc \cong 10Hz$ [25].

O sinal resultante do circuito é analógico e compreendido entre os 0 e $5V$ (uma vez que $Vcc = 5V$), possibilitando a sua leitura pela ADC do microcontrolador. Para além do controlo sobre o potenciómetro, outra forma de alterar a sensibilidade do sensor é ajustando os limiares de decisão (*thresholds*) (ver figura 47) a partir dos quais a deteção é assumida como válida ou não. Assim, níveis de *threshold* mais baixos permitirão que amplitudes de sinal mais reduzidas sejam assumidas como deteção de movimento.

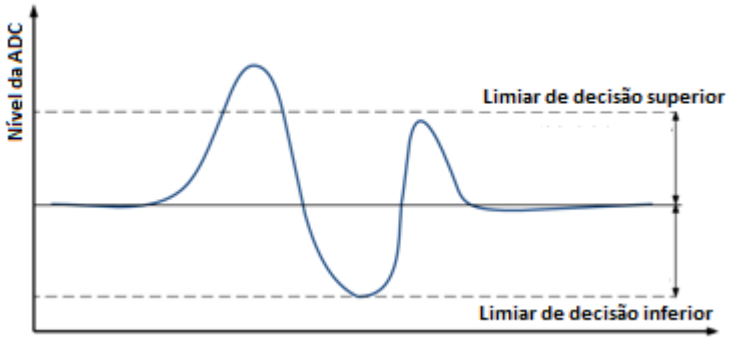


Figura 47 - Exemplo de um sinal analógico para leitura pela ADC.

Neste ponto do trabalho, já se tem ao dispor os dois sensores de movimento a funcionarem de forma independente e dotados com capacidade de serem controlados pelo microcontrolador. Para cumprir um dos objetivos propostos, resta portanto a criação de um sensor híbrido, utilizando estes dois sensores em conjunto. Na figura 48 é demonstrado sucintamente o princípio de funcionamento deste sensor.

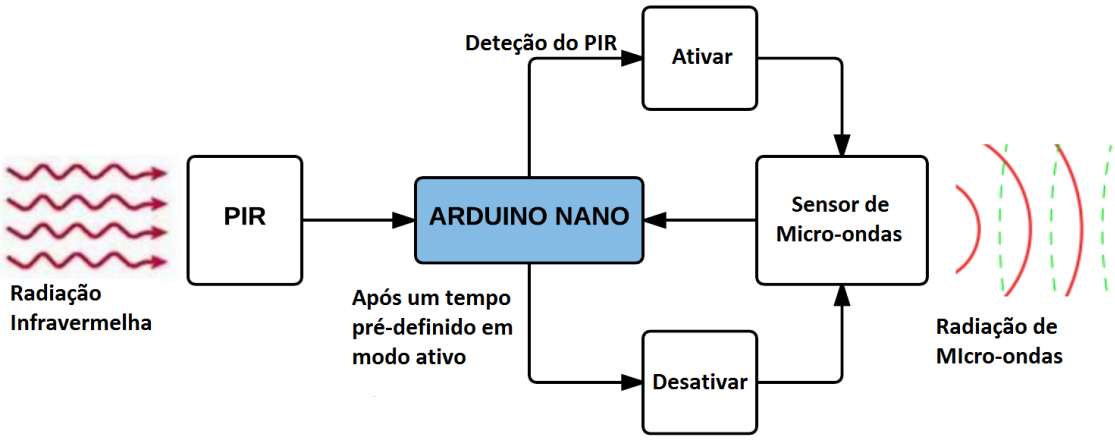


Figura 48 - Princípio de funcionamento do sensor híbrido.

Sendo o PIR um sensor passivo, este estará sempre recetivo à radiação infravermelha proveniente de um corpo ou objeto dentro do seu campo de visão. O sinal é tratado pela ADC do microcontrolador, e caso a amplitude deste atinja o *threshold* programado, é assumida a deteção por parte deste. Tal como foi visto no capítulo sobre sensores, o PIR é muito suscetível a fatores externos e falsos alarmes, existindo por isso, a necessidade de validar essa informação. Para isso, sempre que exista deteção da parte do PIR, o microcontrolador irá ativar o sensor de micro-ondas pondo este em funcionamento durante um tempo pré-estabelecido. Se durante esse tempo o sensor detetar movimento, então é validada a informação de movimento detetado, caso contrário, essa informação é descartada.

4.1.3 – Módulos de comunicação

Os módulos de comunicação desempenham um papel importantíssimo no sistema, uma vez que em caso de falha, impossibilitam não só o envio de dados para o *gateway* mas também a configuração remota. Na figura 49 é possível observar um dos protótipos laboratoriais montados para teste dos módulos $nRF24L01+$, e onde se pode identificar a utilização de dois módulos com antena integrada e um terceiro com antena externa e amplificador. Para interação com estas, utilizaram-se três Arduino Nano.

O dispositivo visto no canto inferior esquerdo da placa, é um regulador de tensão linear de $3.3V$ compatível com *breadboard*, e com a função de alimentar os módulos a

partir do USB do computador, uma vez que o Arduino Nano não tem capacidade para alimentar os módulos através da sua saída de $3.3V$.

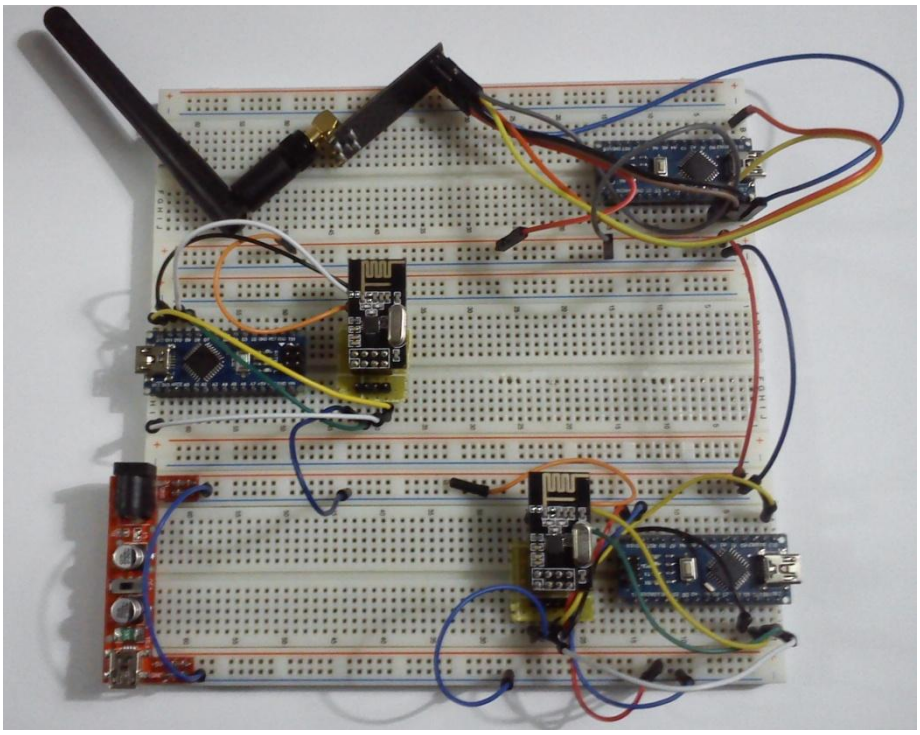


Figura 49 - Protótipo laboratorial para teste dos módulos nRF24L01+.

Este *setup* inicial serviu para testar o funcionamento dos módulos e de algumas das bibliotecas de *firmware* disponíveis. Após a correta comunicação entre duas antenas, procedeu-se à implementação de uma rede em topologia do tipo árvore, com os três módulos a funcionarem em conjunto. Na figura 50, está exemplificada uma possível rede deste tipo, que poderia ser utilizada com recurso a estes módulos de comunicação.

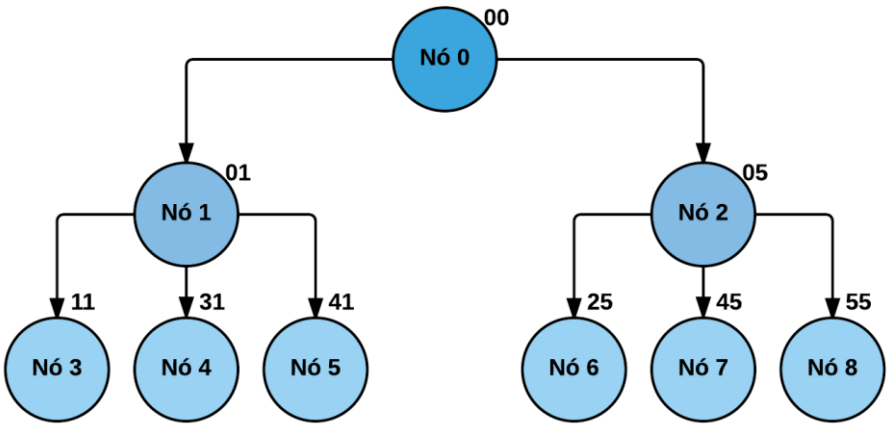


Figura 50 - Exemplo de uma topologia de rede em árvore.

A topologia é constituída por três camadas, tendo no topo o $nó\ 0$, que é o nó base da rede e representa o *gateway* do sistema. No segundo nível temos os denominados nós filhos de primeiro nível e no terceiro, os nós filhos de segundo nível. Para se perceber o funcionamento desta topologia, é importante referir que um determinado nó apenas consegue comunicar diretamente com o seu nó pai. Por exemplo, se o $nó\ 3$ pretender comunicar com o nó da outra extremidade da rede ($nó\ 8$), a informação passará do $nó\ 3$ para o $nó\ 1$, seguindo-se o $nó\ 0$, 2 e finalmente o $nó\ 8$. Mesmo que a comunicação seja entre os nós 4 e 5 por exemplo, a informação passará pelo $nó\ 1$ antes de chegar ao $nó\ 5$.

Os valores que se encontram ao lado dos nós, são exemplos de possíveis endereços (em octal) que estes podem ter numa rede como esta. Para facilitar a compreensão, o exemplo do $nó\ 6$ com o endereço 25, é o segundo filho do $nó\ 2$, que possui o endereço 05, enquanto o $nó\ 8$ será o quinto filho do $nó\ 2$.

4.1.4 – Alimentação

Para alimentar todo o sistema, temos á disposição a tensão da rede ($230VAC/\ 50Hz$), havendo por isso a necessidade de conversão para DC. As tensões necessárias para o funcionamento do sistema são:

Alimentação	Componente/Propósito
$\mathbf{10}\boldsymbol{V}$	$LHi968$, $Arduino\ Nano$ e PWM de controlo do LED driver
$\mathbf{5}\boldsymbol{V}$	$MDU1750$, OpAmps, $AD5241$
$\mathbf{3.3}\boldsymbol{V}$	$ADXL345$, $DHT22$, $nRF24L01+$

Tabela 12 - Alimentações presentes no sistema.

Para esse efeito decidiu-se utilizar um conversor de comutação do tipo $buck$, para baixar a tensão dos $230VAC$ até $12Vdc$. Para a obtenção das tensões da tabela 12, recorreu-se a reguladores de tensão lineares, uma vez que o consumo do sistema é reduzido.

O conversor utilizado é o $LNK306P$, um conversor de custo reduzido e boa eficiência, capaz de responder às necessidades do sistema. O circuito projetado pode ser visto na figura B.1 do Anexo B.

4.1.5 – Protótipo laboratorial final

Após a validação do sensor híbrido e dos módulos de comunicação, desenvolveram-se os drivers para os sensores de temperatura e humidade, pressão e acelerómetro. O desenvolvimento nesta parte do trabalho incidiu quase por completo em

firmware uma vez que os módulos dos sensores que necessitavam de eletrónica adicional, já a continham. Por fim, juntou-se todos os módulos desenvolvidos em *breadboard*[12], criando-se a placa de teste final que pode ser vista na figura abaixo, e que permitirá o desenvolvimento do restante *firmware*.

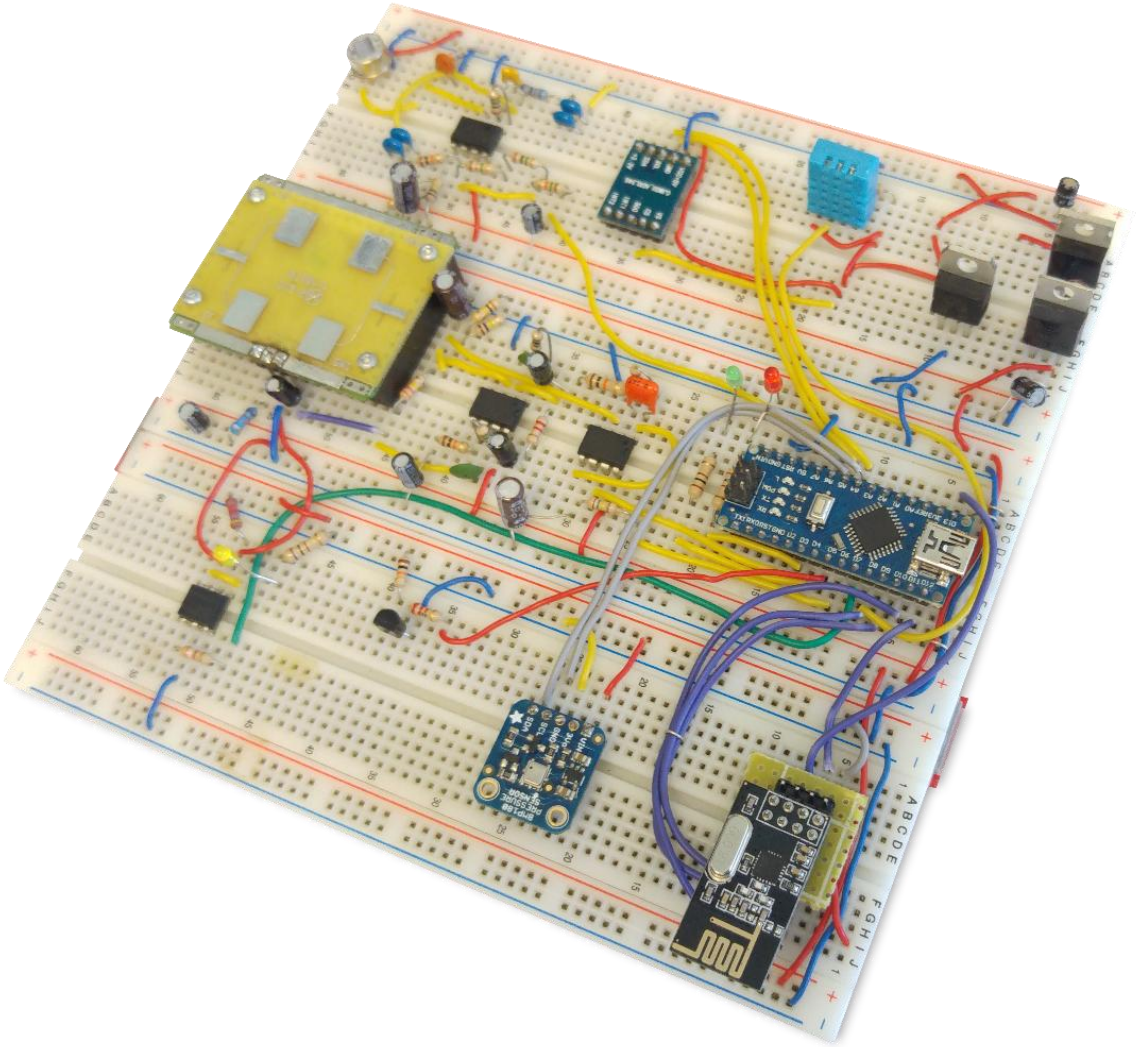


Figura 51 - Placa desenvolvida para teste dos vários módulos.

[12]*Breadboard*, placa branca ou placa de ensaios, é uma placa com orifícios e ligações condutoras, muito usada laboratorialmente para montagem e teste de circuitos.

4.1.6 – Módulo desenvolvido - PCB

Após o restante desenvolvimento estar concluído, procedeu-se ao desenho (que pode ser observado nas secções B1 e B2 do Anexo B), produção e "*assemblagem"* da PCB final que foi colocada na luminária. Na figura 52 é possível observar o resultado final.

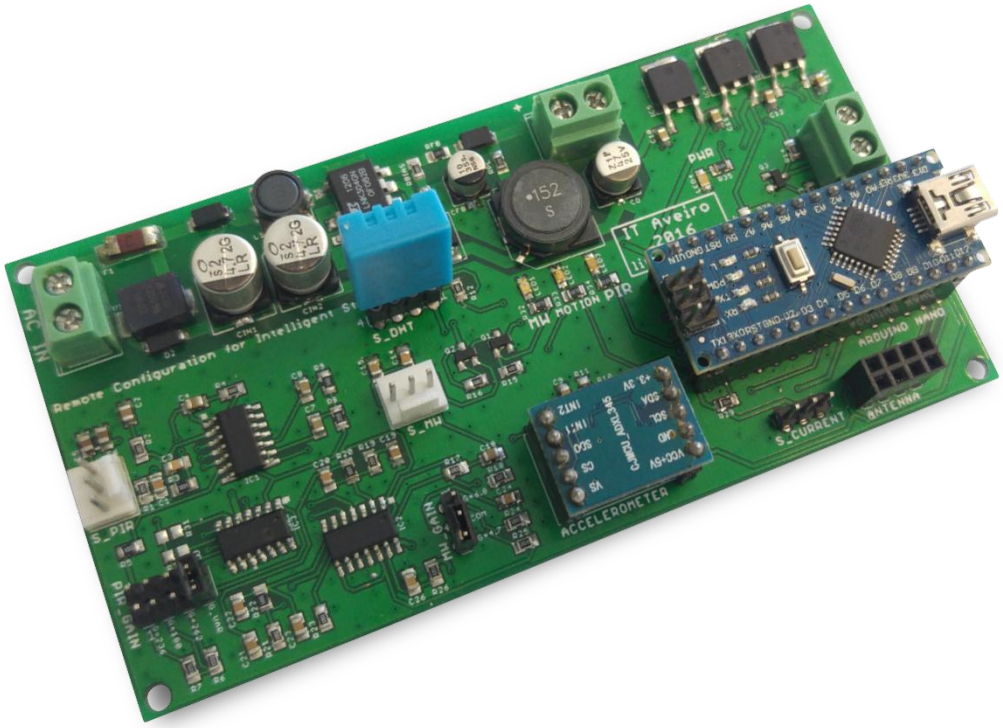


Figura 52 - Módulo da luminária desenvolvido em PCB.

4.1.7 – Montagem da luminária

No presente ponto do trabalho, procedeu-se à montagem final para a obtenção de uma luminária funcional e robusta, capaz de ser implementada num cenário real. Na figura A.1 do Anexo A, é possível verificar a montagem interna da luminária, podendo-se identificar alguns dos seus constituintes como a placa de controlo, o driver para os LED's e o módulo de comunicação.

Na figura A.2 temos uma vista frontal da luminária, podendo-se observar o difusor para os LED's e o sensor híbrido criado anteriormente.

4.2 – Módulo de gestão remota

Um dos objetivos do sistema, é permitir que o responsável pela gestão de determinado sistema de IP, a possa fazer de forma remota, sem necessidade de deslocação ao local onde esta se encontra instalada. No entanto, e porque isso requereria uma alteração drástica no sistema já implementado, numa primeira abordagem, optou-se pela criação de um módulo intermédio com capacidade de monitorização e configuração da luminária, bastando para tal que se encontre no raio de ação do módulo de comunicação.

4.2.1 – Módulo desenvolvido – PCB

O módulo criado pode ser visto na figura 53 e a informação relativa ao seu desenho e produção pode ser consultada nas secções B2 e B3 do Anexo B. O *firmware* desenvolvido para este módulo está presente no Anexo D.

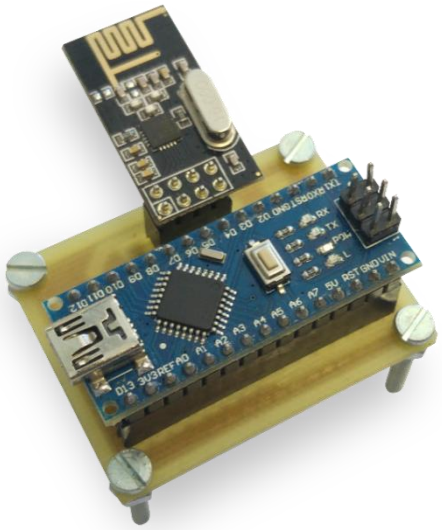


Figura 53 - Módulo de gestão remota.

Este módulo é composto por um Arduino Nano e uma $nRF24L01+$. O *firmware* desenvolvido para este módulo (que pode ser visto no Anexo D) permite interagir de diversas formas com a luminária, sendo a interface com o utilizador o "Monitor Série" do IDE do Arduino ou qualquer outro Terminal Série instalado. De seguida será feita uma apresentação das ações permitidas pelo módulo.

4.2.2 – Modo de funcionamento

Quando se inicia o módulo de gestão remota, é apresentado no terminal o menu da figura 54, com uma lista de operações distintas que podem ser efetuadas.

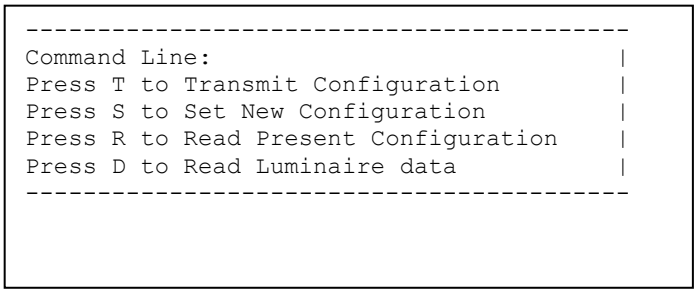


Figura 54 - Menu do módulo de gestão.

Premindo a tecla ‘D’, a luminária envia para o módulo de gestão remota, um pacote de dados com os dados que envia para o *gateway*. Esses dados são apresentados no terminal da forma que pode ser observada na figura 55.

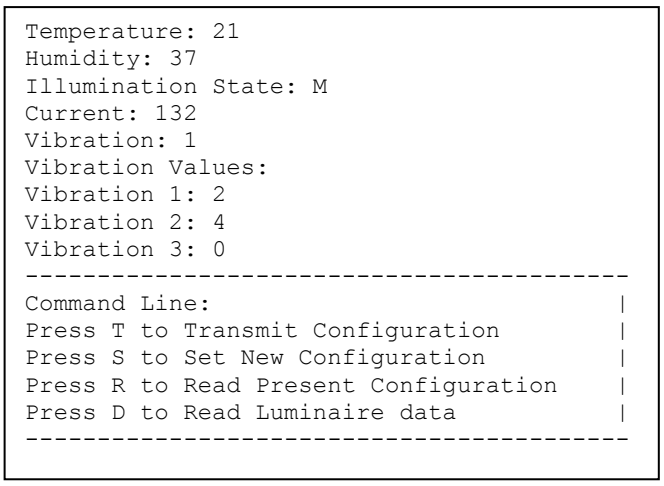


Figura 55 - Dados de monitorização provenientes da luminária.

O pacote de dados que a luminária envia para o *gateway*, contém para além de um cabeçalho de 8 bytes, informação sobre a temperatura dos LED’s e a humidade a que estão sujeitos, o estado de iluminação (‘O’ – off; ‘M’ – máximo; ‘m’ – mínimo e ‘d’ - *dimming*), a corrente consumida e se ocorreu vibração (1 - vibração detetada; 0 - inatividade) na luminária desde a última transmissão, apresentando de seguida os valores de intensidade. Na figura 56, é possível observar a estrutura de dados enviada.

```
struct payload_t {

uint16_t this_node;
uint16_t other_node;
uint16_t packet_id;
uint16_t message_type;

byte temperature;
byte humidity;
boolean vibration;
char illuminationState;
float ef_current;
int vibration1;
int vibration2;
int vibration3;
};
```

Figura 56 - Pacote de dados enviado para o *gateway*.

Para se configurarem novos parâmetros, premindo a tecla ‘S’ aparecerá um menu onde é pedido a configuração de vário parâmetros do sistema.

Após a validação do valor inserido, novo parâmetro será pedido e continuar-se-á até terminarem os parâmetros de configuração, obtendo-se uma janela como na figura 57.

```
-----------------------------------
************* Configuration Menu *************
-------------------------------------------
--------- Light Profile Configuration --------
MaxLight(0-100%): 100
MinLight(0-100%): 10
MaxTime(sec): 120
DimmTime(sec): 5
MinTime[0 -> never off](sec): 0
-------------------------------------------
------------- Sensors Configuration ----------
PIR:
Gain(0-100%): 100
Thresh(60-100%): 70

MW:
Thresh(60-100%): 80
Npulses: 3

ADXL:
Thresh: 40

Data acquisition(sec): 10
```

Figura 57 - Janela de configuração totalmente preenchida.

Da análise da figura, é possível ver os parâmetros da luminária que são configuráveis através da placa de gestão remota.

Em relação ao perfil de iluminação, é possível escolher a intensidade máxima ($MaxLight$) (que surgirá quando ocorrer deteção de movimento) e a mínima ($MinLight$). Além disso, é possível escolher o tempo que se pretende que a iluminação esteja na sua intensidade máxima ($MaxTime$), em dimming ($DimmTime$) e na intensidade mínima ($MinTime$), havendo a possibilidade que este tempo seja *"infinito"* isto é, que a luminária nunca se apague.

Relativamente aos sensores, no PIR, é possível atuar sobre o potenciómetro digital referido na secção 4.1.2 sobre o sensor híbrido, alterando o seu valor ($Gain$), bem como nos níveis de *threshold* ($Thresh$) do sinal de entrada na ADC, possibilitando assim ajustar estes dois parâmetros de forma a se obter o melhor resultado possível.

Quanto ao sensor de micro-ondas é possível alterar os níveis de deteção, fazendo variar o *threshold* ($Thresh$). Esse valor irá alterar a tensão de referência no OpAmp do estágio de comparação, tornado o sensor mais ou menos sensível mediante a alteração efetuada. Além do parâmetro referido, pode-se também definir a quantidade de pulsos detetados para se validar a deteção, sendo que o número de pulsos está diretamente relacionado com as alterações efetuadas no nível de *threshold* do sensor de micro-ondas (ver figura 45).

É também possível configurar a sensibilidade do sensor de vibração (acelerómetro), alterando o seu valor de *threshold*. Finalmente, a última configuração pedida é o período de transmissão das mensagens de monitorização para o *gateway*.

Após a configuração, para se enviar essa informação basta premir a tecla 'T' onde aparecerá a mensagem da figura 58, pedindo para o utilizador inserir o número da luminária que pretende configurar (no caso da figura, seria a luminária 11).

```
Send to Luminaire #: 11
Sending Configuration...
```

Figura 58 - Processo de envio da configuração.

Caso a luminária receba o pacote com as novas configurações, esta irá acender e apagar três vezes, dando assim *feedback* visual ao utilizador, da conclusão do processo de configuração.

Para se obter informação sobre os valores de configuração presentes na luminária, é possível fazer um pedido de leitura das configurações em utilização premindo a tecla 'R'. Uma mensagem igual à da figura 58 aparecerá e após selecionada a luminária pretendida, o resultado obtido será como o mostrado na figura 59.

```
---------------------------------
Configured parameters:
MaxLight(%): 100
MinLight(%): 10
MaxTime(sec): 120
DimmTime(sec): 5
MinTime(sec): 0
PIR Gain(%): 100
PIR Thresh(%): 70
MW Thresh(%): 80
MW nPulses: 3
ADXL Thresh: 40
Data acquisition(sec): 10
---------------------------------
```

Figura 59 - Leitura dos parâmetros configurados na luminária.

4.3 – Resultados experimentais

Após a montagem da luminária, a criação do módulo de gestão remota e finalizados os respetivos *firmwares*, tornou-se possível o teste de todo o sistema desenvolvido a trabalhar em conjunto. Procurou-se efetuar os testes num cenário o mais próximo possível do real. Para isso, montou-se a luminária a uma altura de $3.35\,m$ e desenhou-se no chão uma grelha em redor da posição do sensor híbrido, com uma área de 2.5 por $2.5\,m$ e espaçamento de $0.5\,m$ entre linhas. Nas figuras seguintes é possível observar alguns dos dados obtidos neste teste. Os gráficos ilustram as secções (de $0.5 \times 0.5\,m$) criadas no chão pela grelha e onde se pode verificar as zonas de cobertura/deteção dos sensores de movimento. Os testes foram repetidos para várias configurações com a alteração de parâmetros, que eram alterados através do módulo de gestão remota desenvolvido. Neste ponto, foi possível verificar a rapidez e facilidade na configuração que este novo sistema possui, uma vez que não houve necessidade de aceder à luminária de cada vez que se pretendia alterar determinado parâmetro. Os testes foram realizados em ambiente *indoor* a uma temperatura de $23℃$.

- ## Testes ao sensor PIR

Numa primeira fase, testou-se o sensor PIR isoladamente, isto é, sem o auxílio do sensor de micro-ondas. Relativamente a este sensor, é possível a alteração de dois parâmetros, o denominado ganho, que consiste na atuação sobre o potenciómetro digital ($0\% \Rightarrow \approx 0\ \Omega$ e $100\% \Rightarrow 1\ M\Omega$), onde a sua variação se refletirá numa alteração do valor

do ganho global do circuito de acondicionamento de sinal do sensor. O outro parâmetro configurável é o nível de *threshold* do sinal lido pelo microcontrolador (rever secção 4.1.2 na parte relativa ao sensor PIR), concebendo-se assim uma maior ou menor sensibilidade ao sensor.

Inicialmente manteve-se o ganho constante ($0\% \Rightarrow 0\Omega$), fazendo-se variar os níveis de *threshold*. Obtiveram-se os seguintes resultados:

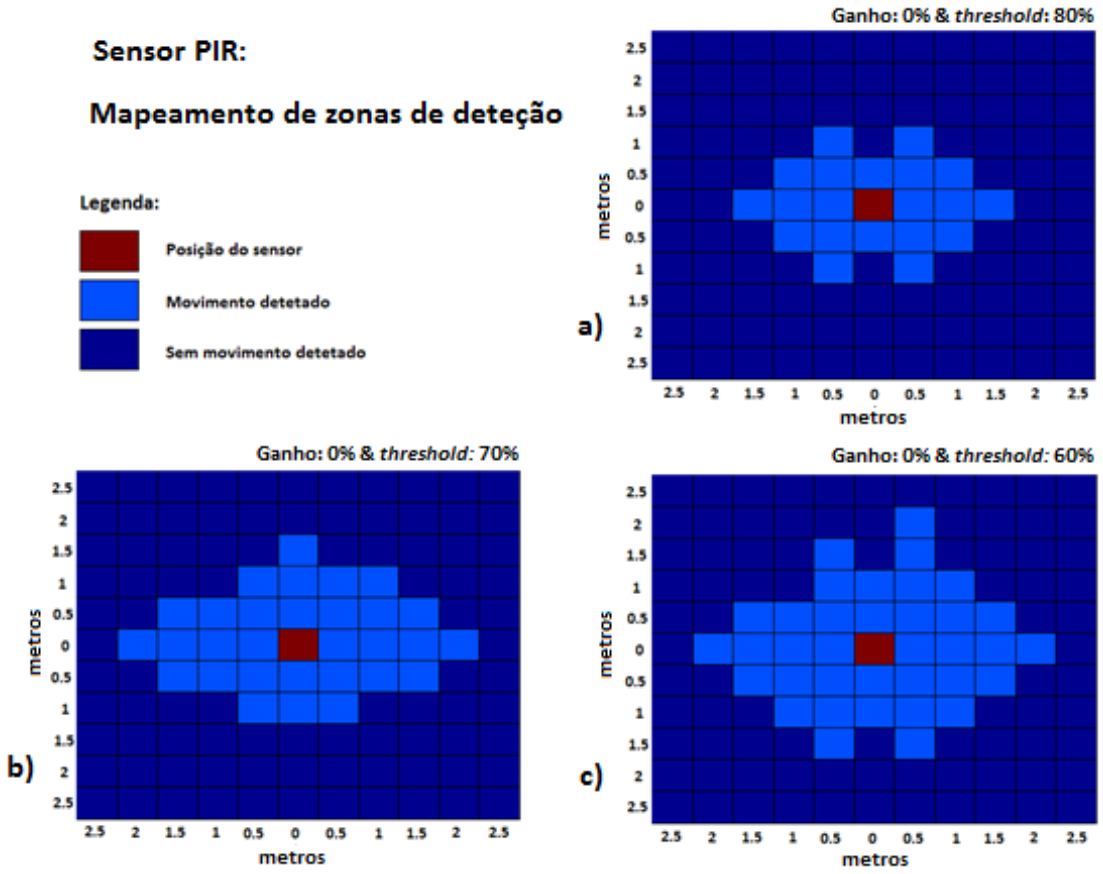


Figura 60 - Mapeamento de deteção de movimento utilizando PIR e com variação dos níveis de *threshold*.

Na fase seguinte, manteve-se o nível de *threshold* constante (60%) e variou-se o valor do ganho, obtendo-se o seguinte mapeamento:

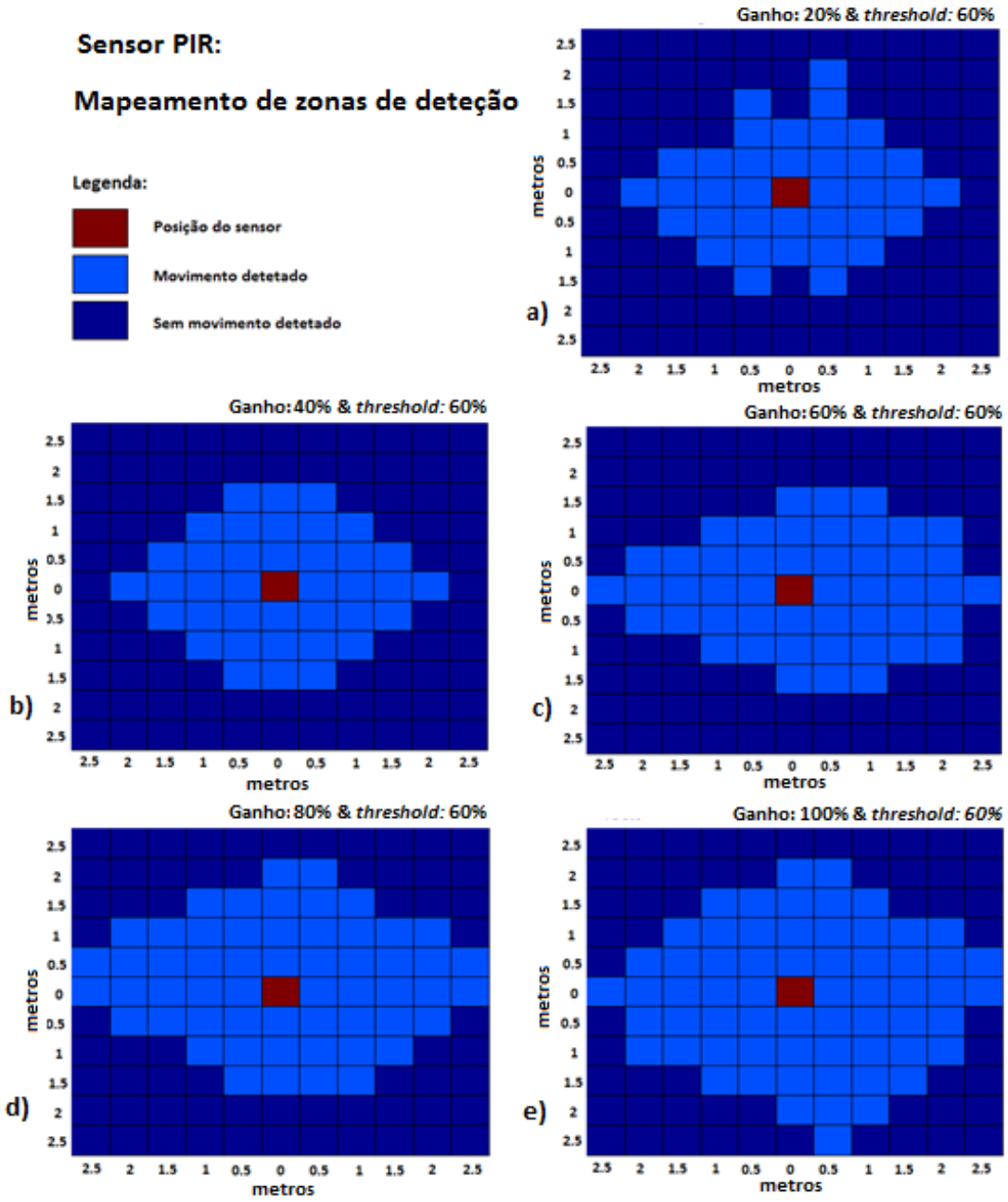


Figura 61 - Mapeamento de deteção de movimento utilizando PIR e com variação do ganho.

- ## Testes ao sensor de MW

Após o término dos testes ao sensor PIR, efectuaram-se de forma análoga, os testes ao sensor de micro-ondas. Tal como no caso anterior, é possível variar dois parâmetros, o *threshold* e o número de pulsos lidos pelo microcontrolador para validação de movimento (rever secção 4.1.2 na parte relativa ao sensor de micro-ondas). Face à

boa estabilidade do sensor, optou-se por manter o número de pulsos reduzido e constante ($Npulses = 2$), embora possa ser alterado se assim se entender, e fez-se variar os níveis de *threshold*. Obteve-se os seguintes resultados:

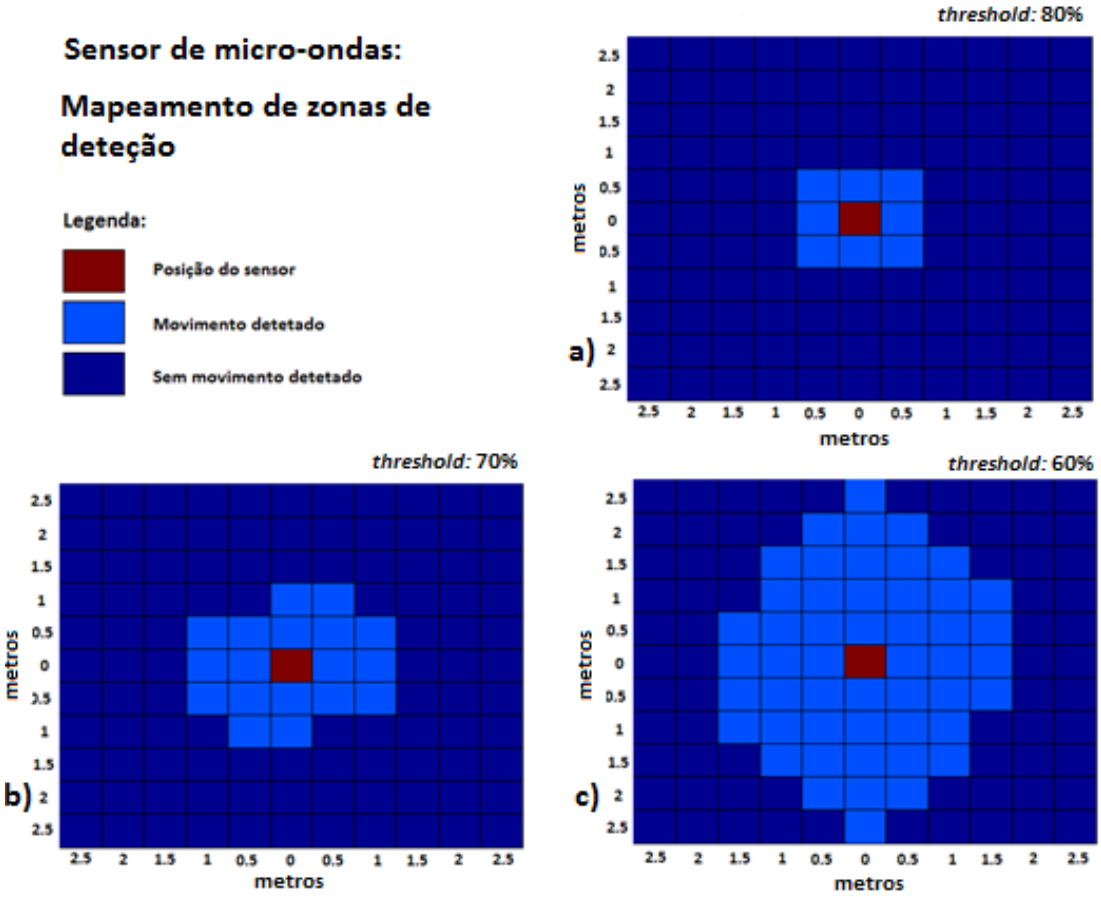


Figura 62 - Mapeamento de deteção de movimento utilizando sensor de micro-ondas e com variação dos níveis de *threshold*.

- ## Testes ao sensor híbrido

Finalmente, procedeu-se ao teste do sensor híbrido, com o PIR e o sensor de micro-ondas a trabalharem em conjunto. Efectuaram-se dois conjuntos de configurações distintas, um mais sensível e outro menos, cujos resultados podem ser observados na figura:

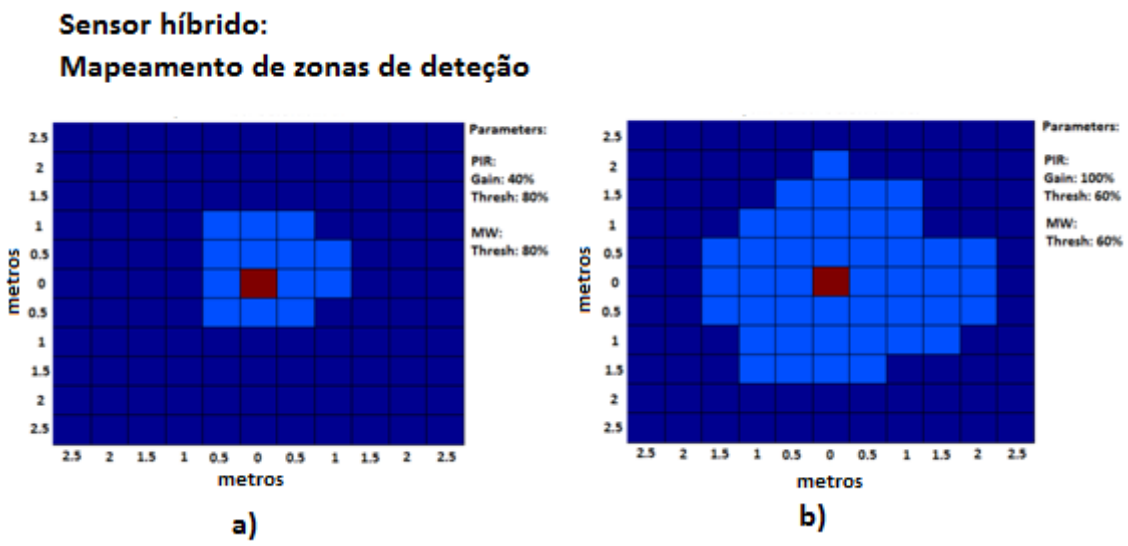


Figura 63 - Mapeamento de deteção de movimento utilizando o sensor híbrido para dois conjuntos de configurações distintos.

- ## Caracterização energética da luminária

Sendo este um sistema que visa a poupança energética, é importante saber o consumo que a luminária desenvolvida apresenta, nos diversos estados que terá em funcionamento. Assim sendo, mediu-se a corrente consumida por esta para os possíveis valores de fluxo luminoso. Na figura abaixo é possível observar o resultado obtido.

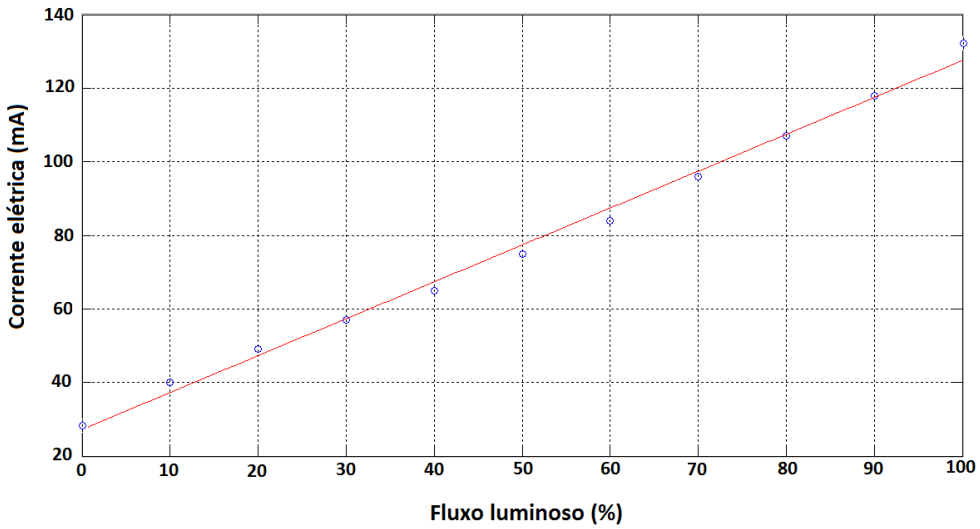


Figura 64 - Caracterização energética da luminária desenvolvida.

Relativamente ao gráfico da figura 64, o consumo verificado para 0%, isto é, na ausência de luz foi de $28\ mA$, o que representa o consumo da placa desenvolvida. Para o valor máximo de iluminação (100%) verificou-se um consumo de $132\ mA$ e para o mínimo (10%), $40mA$.

- ## Conclusões

Após a conclusão dos testes laboratoriais, ficou evidenciado para além da já referida comodidade possibilitada pelo módulo de gestão remota, o grande impacto que a alteração dos diversos parâmetros tem na deteção de movimento. Como se pode observar nos gráficos de mapeamento obtidos, as alterações efetuadas permitiram a variação da sensibilidade de cada um dos sensores (e consequentemente do sensor híbrido). Esta variação tem uma grande importância para o bom funcionamento do sistema, uma vez que permitirá o ajuste do sensor híbrido quando se estiver na presença de condições atmosféricas adversas, como temperaturas elevadas que afetam o PIR ou ventos e chuvas fortes que podem atraiçoar o sensor de micro-ondas.

Desta forma e face aos resultados positivos obtidos, poder-se-á proceder à instalação da luminária criada, no piloto da ponte do Crasto para se dar início à fase seguinte de testes, neste caso, num cenário real.

Capítulo 5

5 – Implementação LITES

5.1 – Sistema

Na figura 65 é possível observar um esquema geral de funcionamento do piloto que se encontra instalado na ponte pedonal do Crasto.

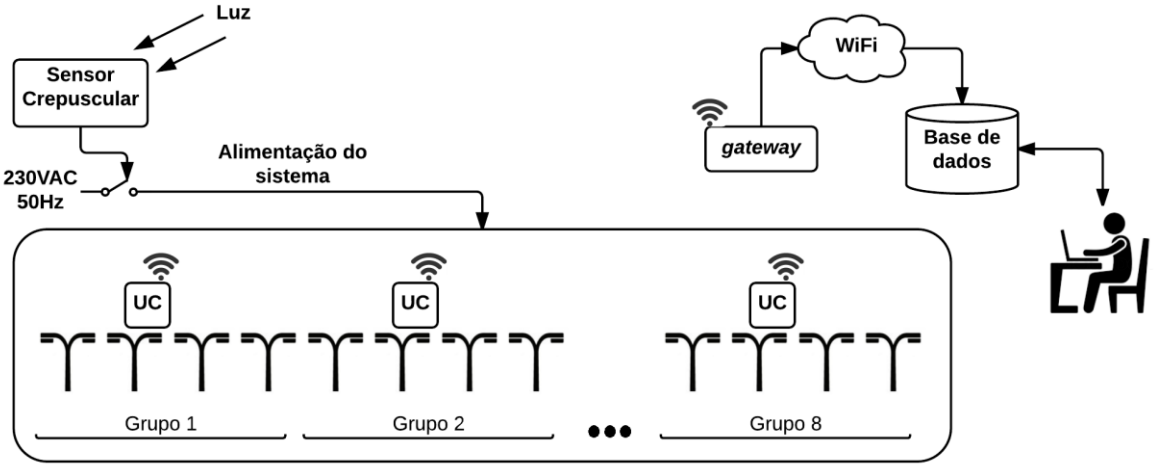


Figura 65 - Esquema global do piloto instalado na ponte pedonal do Crasto.

O piloto é constituído por 31 luminárias que se encontram agrupadas em secções de 4 luminárias (sendo uma delas de apenas 3), numa topologia centralizada, celular e com controlo ativo de energia. A alimentação necessária para o sistema é controlada por um sensor crepuscular, permitindo assim aumentar a eficiência, uma vez que na presença de iluminação natural considerada suficiente, todo o sistema estará desligado e consequentemente, não haverá qualquer consumo de energia.

Dentro de cada grupo, existem luminárias com diferentes funções (para além de iluminação), como comunicação com o *gateway* e deteção de movimento. São as luminárias dotadas de sensores de movimento as responsáveis por controlar a iluminação do próprio grupo e grupo seguinte. Em todos os grupos existe uma luminária dotada de 2 sensores, permitindo assim detectar também o sentido do movimento, possibilitando ativar o grupo seguinte.

As luminárias que possuem módulo de comunicação, enviam a informação recolhida para o *gateway* que posteriormente os remete para uma base de dados que pode ser acedida através de uma interface *web*.

5.2 - *Gateway*

O gateway é o elemento do sistema responsável por transmitir para a base de dados, toda a informação de interesse proveniente das luminárias. É composto por um $BeagleBone\ Black$ (BBB) e um módulo de comunicação $nRF24L01+$ com amplificador, já apresentado neste documento. Relativamente ao BBB, é uma placa de desenvolvimento com enormes recursos e capacidade de correr sistemas operativos como $Ubuntu$, $Debian$ ou $Android$. Na tabela 5.1 são apresentadas algumas das características desta placa.

Processador	$AM3359$ $Sitara\ ARM\ Cortex-A8$
Frequência máxima	$1\ GHz$
Tensão de operação	$3.3V$
Pinos digitais	63
Pinos analógicos	7
RAM	$512MB\ DDR3$
EEPROM	$256\ KB$
Comunicação	$UART, SPI, I^2C, CAN\ Bus$
Dimensões	$8.6\ \times\ 5.3\ cm$

Tabela 5.1 – Características do $BeagleBone\ Black$ [26].

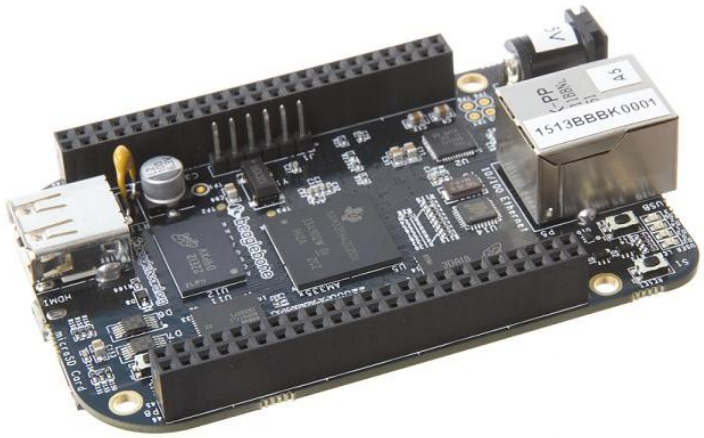


Figura 66 - BeagleBone Black [26].

5.3 – Interface *web*

A interface *web* foi desenvolvida pelo $ATNoG$[13]. Esta permite a visualização gráfica dos dados recolhidos e armazenados na base de dados. Na figura 67 é possível observar os últimos dados recebidos e guardados na base de dados relativos à luminária 11.

Node 11

Node:	name	Description	Process list	last updated	value
11	Temperatura			inactive	52.0
11	Humidade			inactive	11.0
11	Valor_Vibracao1			inactive	3149
11	E_corrente_cal		log	inactive	191
11	Valor_Vibracao3			inactive	1607
11	Valor_Vibracao2			inactive	5699
11	Estado_Luminaria			inactive	-1.00
11	E_corrente		log	inactive	191
11	Vibracao			inactive	8.00

Figura 67 - Dados relativos à luminária 11 observados na interface Web.

Caso se pretenda, é também possível a observação da variação de determinada variável ao longo do tempo. Na figura 68 é possível analisar a variação da temperatura dos LED's ocorrida durante a última hora de monitorização na mesma luminária.

Figura 68 - Gráfico da variação da temperatura numa luminária.

[13] ATNoG (do inglês, Advanced Telecommunications and Network Group) é um grupo de investigação do Instituto de Telecomunicações da Universidade de Aveiro.

5.4 – Resultados obtidos

Nesta secção serão apresentados alguns dos resultados obtidos no piloto de testes. Primeiramente, será feita uma comparação entre o mapeamento das zonas de deteção de movimento da luminária desenvolvida e a já existente, bem como uma comparação do registo de actividade entre as luminárias anteriormente implementadas e a nova luminária desenvolvida. Finalmente, serão mostrados alguns dos dados recolhidos pelo *gateway*, nomeadamente valores de temperatura, humidade e vibração, que podem ser observados através da plataforma Web anteriormente apresentada.

- **Mapeamento de zonas de deteção de movimento**

Na figura seguinte é possível observar as zonas de deteção de movimento da luminária anteriormente utilizada. As luminárias já existentes foram anteriormente configuradas (manualmente) com o ajuste de parâmetros que melhores resultados apresentou para este cenário. Para efeitos de comparação, os mapeamentos apresentados para a luminária desenvolvida, foram efetuados com as duas configurações que melhores resultados apresentaram e que podem ser vistas na figura 69.

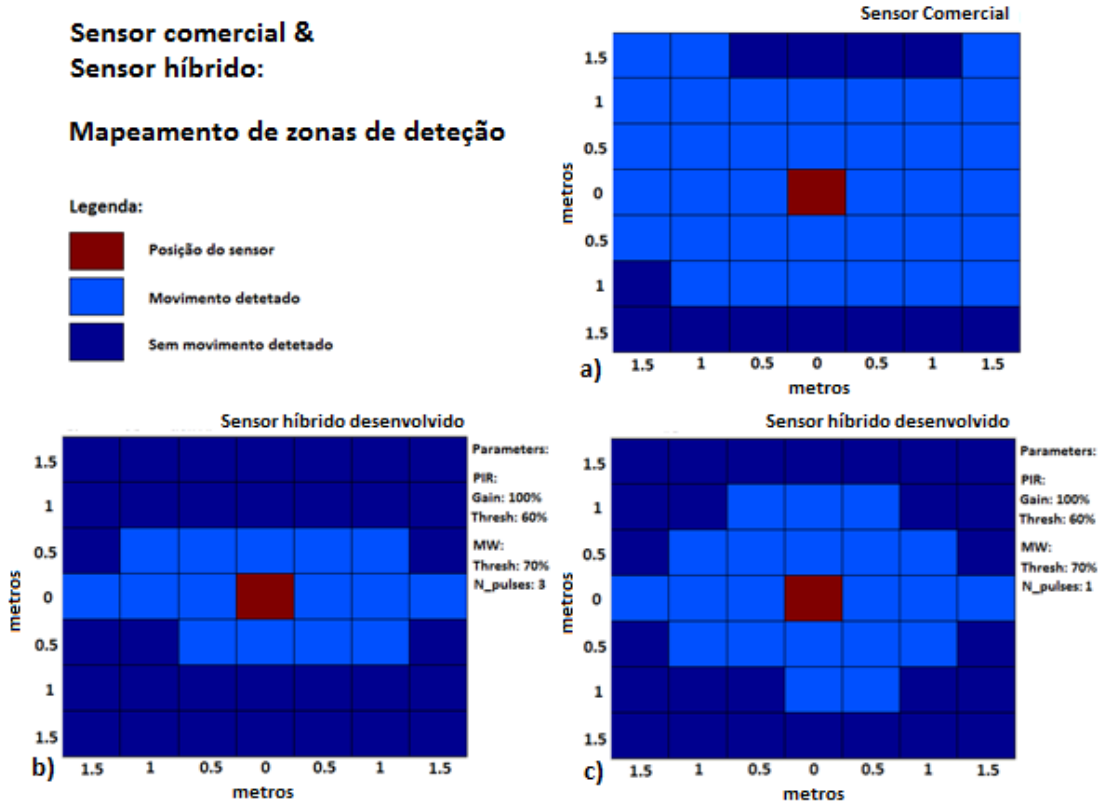


Figura 69 - Comparação de mapeamentos de deteção entre sistemas.

Observando e comparando as figuras acima, conclui-se desde logo que a luminária com sensor comercial é aquela que apresenta melhores resultados, uma vez que a sua área de cobertura é superior quando comparada com as apresentadas pela luminária que utiliza o sensor híbrido desenvolvido. No entanto, também é possível observar que em qualquer uma das configurações, o raio de ação do sensor cobre toda a área de passagem da ponte (tem aproximadamente $3.1\ m$ de largura). Assim sendo, em cenários onde a topologia utilizada seja centralizada (como é o caso do sistema implementado na ponte do Crasto), os mapeamentos obtidos não trazem qualquer inconveniente uma vez que a atuação sobre a iluminação (após a deteção feita por um sensor) será feita nas luminárias seguintes e não na própria.

- ## Registo comportamental das luminárias

Através das figuras seguintes, obtidas através da plataforma *Web* apresentada anteriormente, é possível observar o comportamento de 3 luminárias distintas durante um determinado período temporal. As luminárias 4 e 7 fazem parte do sistema já implementado, enquanto a luminária 11 contém o módulo desenvolvido no presente trabalho. O período temporal em observação situa-se entre as 16:00 horas do dia 19 de Novembro de 2016 até às 08:00 horas do dia 20 de Novembro de 2016. O motivo da data escolhida prende-se pelas condições climatéricas adversas que se fizeram sentir, nomeadamente chuva, ventos fortes e trovoada.

Os gráficos mostram o estado da luminária ao longo do tempo, sendo que o valor 100 corresponde à luminária na sua intensidade máxima, 50 durante o processo de *fade-out*, 10 na intensidade mínima e 0 quando desligada.

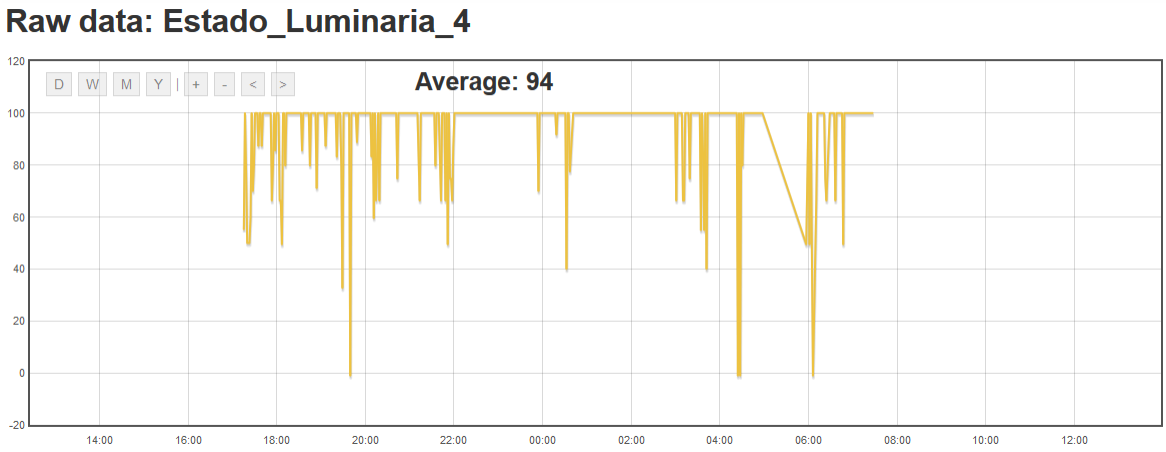


Figura 70 - Comportamento da luminária 4.

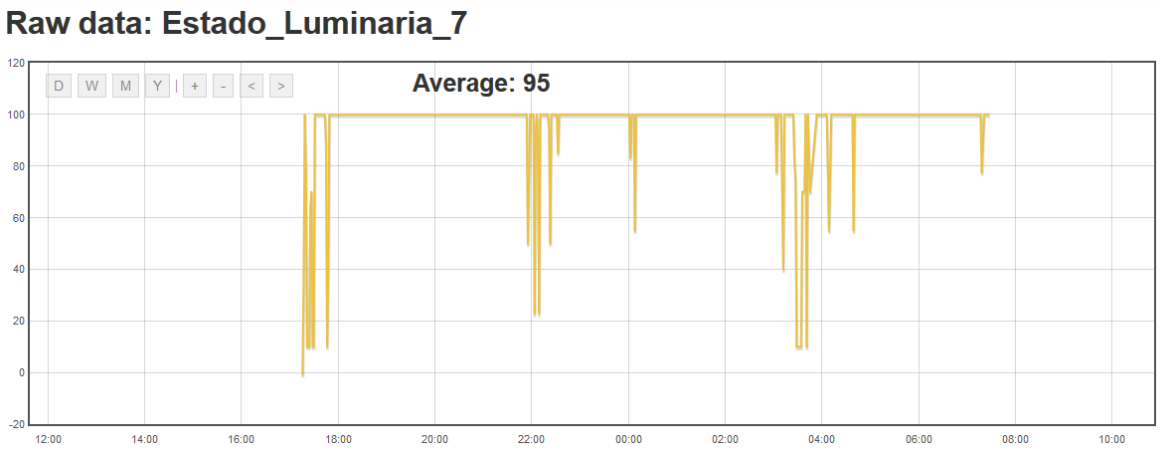


Figura 71 - Comportamento da luminária 7**.**

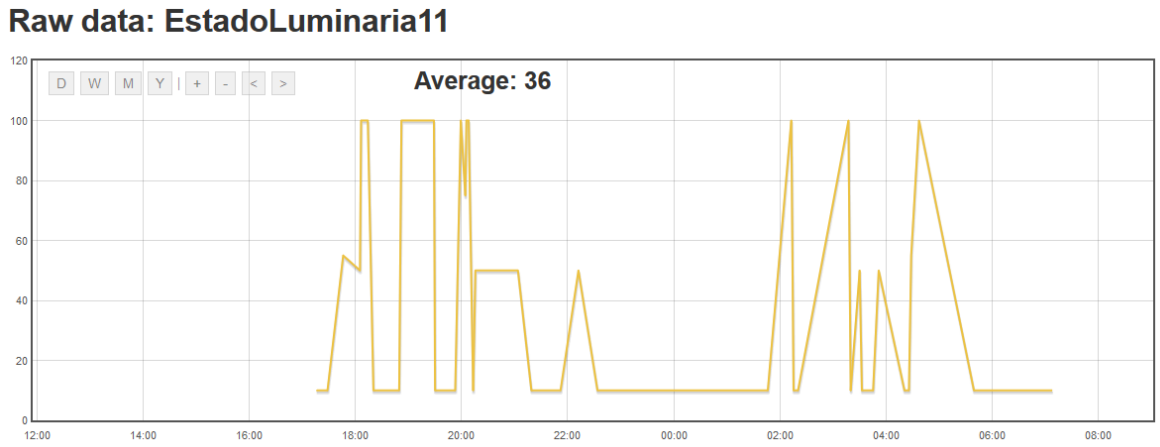


Figura 72 - Comportamento da luminária 11.

Da análise dos gráficos acima, verificou-se que as luminárias apenas entraram em funcionamento depois das 17:00 horas e assim permaneceram até perto das 07:00 horas do dia seguinte, correspondendo ao período noturno. Isto ocorre devido ao sensor crepuscular utilizado, que apenas permite que o sistema seja alimentado quando a luz natural existente não for suficiente (figura 65).

Outro dado relevante que se retira da análise das figuras, é a diferença no tempo em que as luminárias estiveram na sua intensidade máxima. As duas luminárias com o sensor de micro-ondas comercial, estiveram praticamente a noite toda no seu estado máximo, enquanto que a luminária desenvolvida com a utilização do sensor híbrido, apresentou um comportamento bem distinto, tendo sido disparada apenas algumas vezes, demonstrando uma melhoria significativa nesse aspecto.

- **Monitorização de variáveis**

Nas figuras seguintes é possível observar o comportamento das outras variáveis do sistema durante a janela temporal mencionada anteriormente. A variação da temperatura, humidade, consumo de corrente ou valores de vibração pode ser vista igualmente através da plataforma.

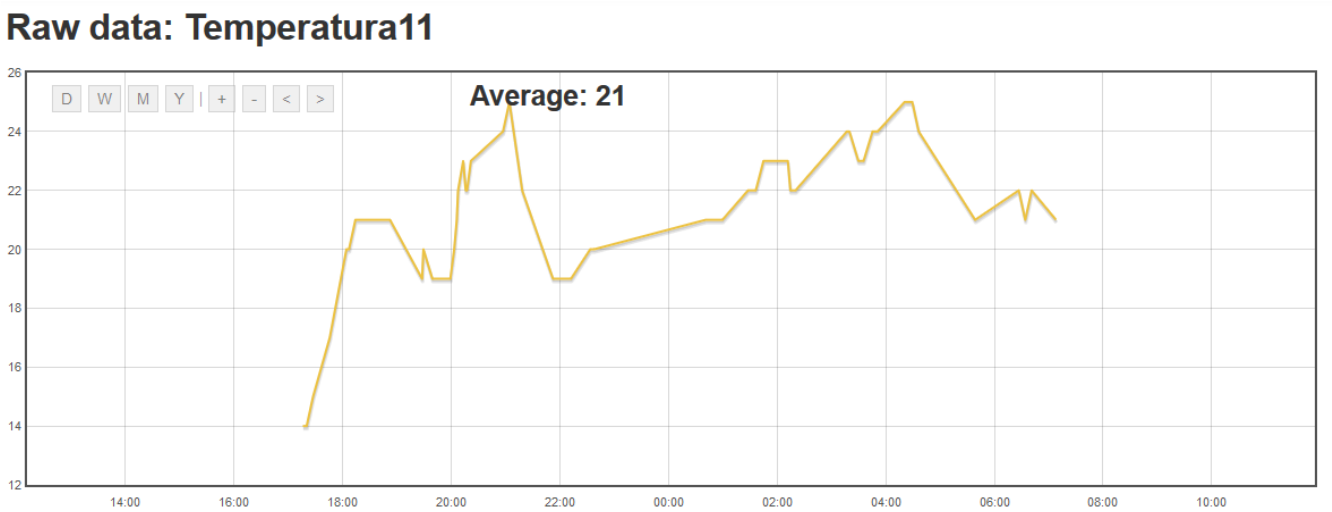


Figura 73 - Variação da temperatura na luminária 11.

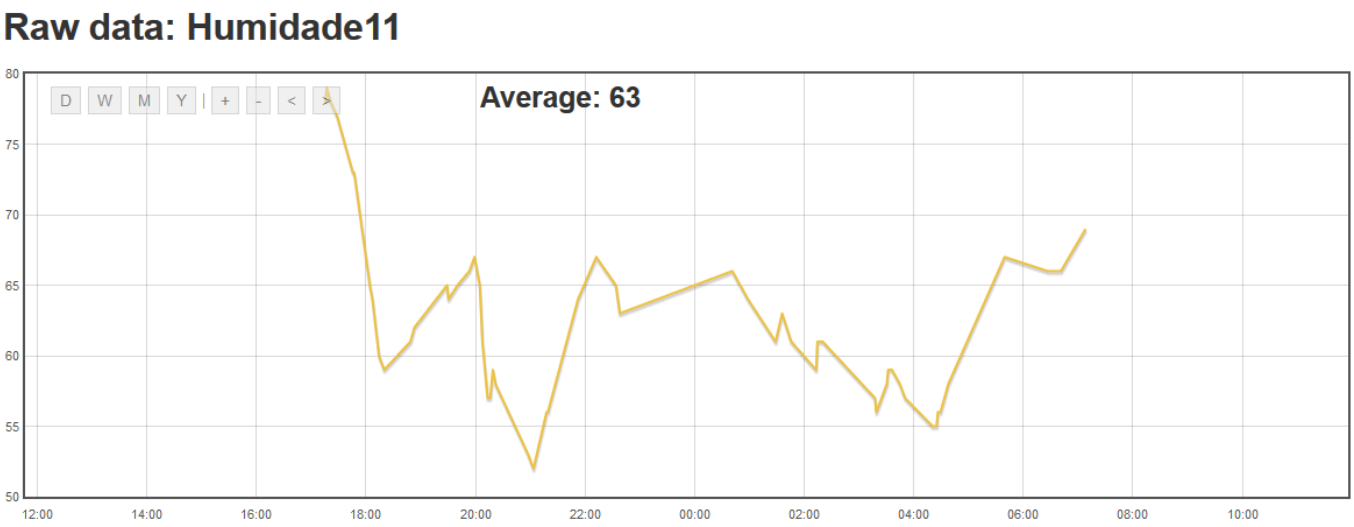


Figura 74 - Variação da humidade na luminária 11.

Através da comparação entre as figuras 72 e 73, é possível observar a relação entre o estado da luminária e a temperatura nos LED's. Os picos de temperatura obtidos surgem em janelas temporais onde a iluminação esteve no seu estado máximo, o que era expectável, uma vez que é nessa situação que os LED's apresentam o seu consumo máximo, dissipando mais energia sob a forma de calor.

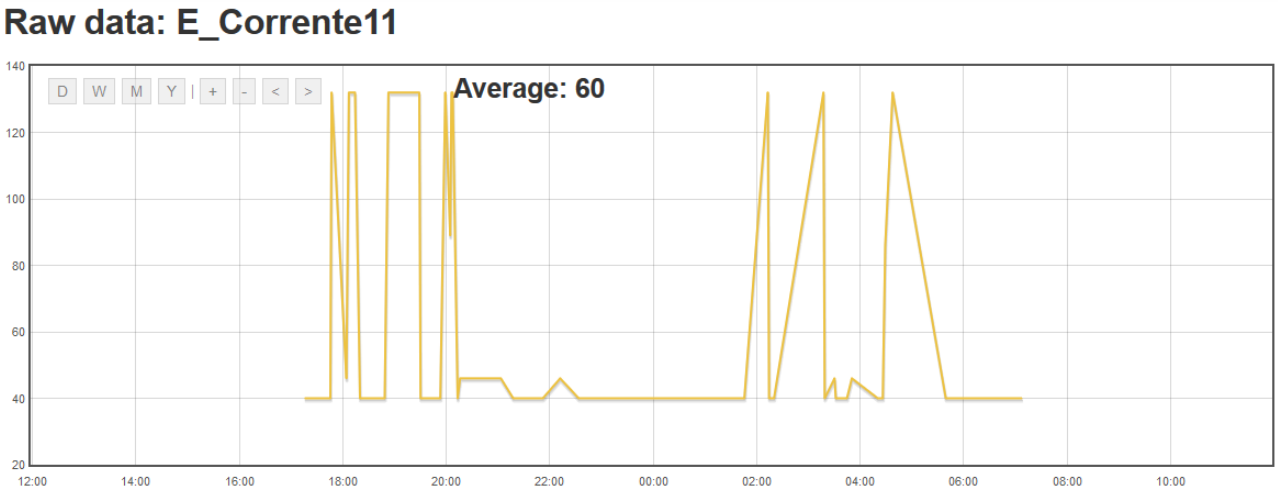


Figura 75 - Variação do consumo de corrente da luminária 11.

Comparando a figura 75 novamente com a figura 72, é evidente a relação entre ambas, uma vez que têm um comportamento idêntico. Tal seria de esperar pois o consumo está diretamente relacionado com o estado que a luminária apresenta. Como podemos observar, os picos de consumo ocorreram quando a luminária estava no seu estado máximo, isto é, com o seu fluxo luminoso no nível máximo.

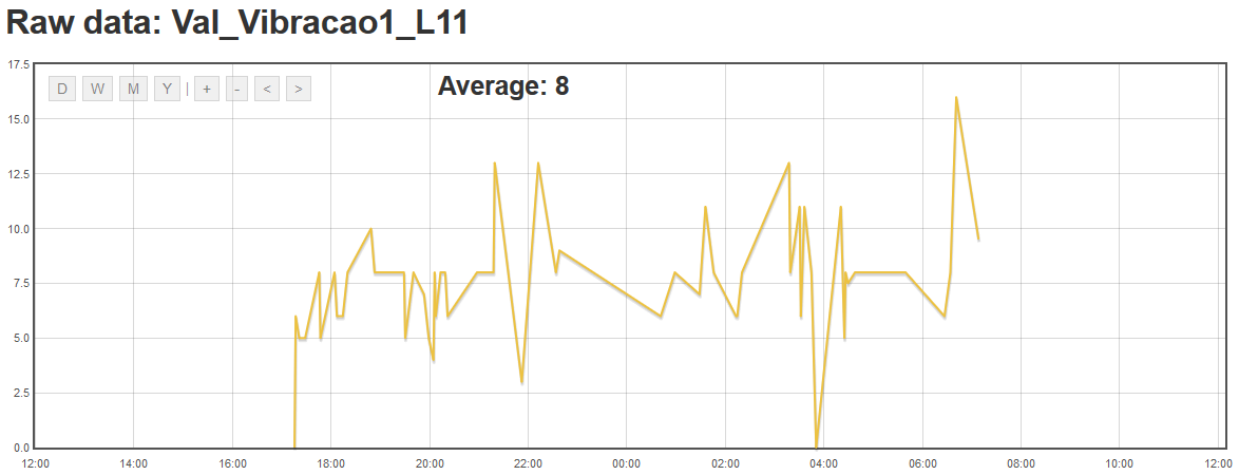


Figura 76 - Valores de vibração detetados na luminária 11.

No gráfico 76 é possível observar os valores de vibração detectados na ponte, durante a janela temporal em análise. Tal como referido anteriormente, as condições climatéricas a que a luminária esteve sujeita foram agressivas, com chuva e ventos fortes, que se fizeram sentir na estrutura da ponte, como se pode ver na figura acima onde a vibração foi uma constante ao longo desse período.

Capítulo 6

6 – Análise final

6.1 – Discussão e conclusões gerais

Após concluído o trabalho, é tempo de se fazer um balanço do que foi feito e discutir os resultados obtidos.

Relativamente aos objetivos inicialmente propostos, em especial à criação de um sensor híbrido e olhando para os resultados obtidos e mostrados nas secções 4.3 e 5.4, pode-se concluir que esta solução aparenta ter a viabilidade necessária para ser tida em conta como solução para deteção de movimento em sistemas de IP. Os resultados alcançados na primeira fase de testes em cenário real são animadores e o novo sensor aparenta ter um bom comportamento com condições atmosféricas adversas. Há no entanto, a necessidade de se testar o protótipo do sensor híbrido durante mais tempo em cenário real e condições atmosféricas adversas, para que mais conclusões se possam tirar, uma vez que devido ao curto espaço de tempo entre a sua implementação e a escrita deste documento, não foi possível fazer testes exaustivos ao sensor.

Os restantes objetivos estavam relacionados diretamente com a criação de um módulo de controlo, que permitisse a aquisição de dados e posterior envio para o *gateway,* bem como ser dotado de capacidade de configuração remota.

O módulo criado e demonstrado no presente documento, possui todas as características propostas, e o seu funcionamento com o módulo de gestão remota também desenvolvido durante o presente trabalho é muito bom, embora após a aplicação no piloto de testes na ponte do Crasto, alguns problemas tenham surgido. Em relação ao sensor híbrido, e devido ao facto da composição do material constituinte da tampa da luminária filtrar a radiação infravermelha, este teve que ser colocado fora da estrutura da luminária, impedindo o fecho completo da tampa, comprometendo o bom funcionamento do sistema. Assim como pelo facto da caixa do sensor ser bastante mais baixa que toda a estrutura da luminária, esta acaba por obstruir o FoV do sensor em algumas direções, comprometendo assim melhores resultados. No que diz respeito à comunicação, também aqui se observaram alguns problemas que não se verificaram aquando dos testes laboratoriais. Esses problemas derivam da saturação que a rede apresenta, criando um grande número de colisões e consequentemente, causando a perda de inúmeros pacotes, dificultando assim a comunicação.

Como se acabou de ver, o sistema atualmente instalado bem como o módulo apresentado ao longo desta dissertação, possuem algumas debilidades que devem ser resolvidas futuramente. Há no entanto indicações, de que o trabalho efetuado não só acrescentou valor ao sistema desenvolvido anteriormente, como foi um passo importante para dotar o sistema de configuração remota.

6.2 – Trabalho futuro

Olhando para o resultado final da junção entre o sistema em utilização e os módulos desenvolvidos, verifica-se que este tem um potencial enorme para ser trabalhado e melhorado.

Numa primeira fase, seria importante a validação do sensor híbrido em cenário real durante um maior período de tempo, permitindo validar o seu correcto funcionamento ou a deteção de falhas a serem corrigidas. Além disso, um estudo sobre o seu comportamento com a variação das condições atmosféricas, permitiria programar a unidade controlo de tal forma que esta reconfigurasse automaticamente os sensores mediante as condições verificadas. Para isso, seria também necessário dotar o sistema de mais alguns sensores, que permitissem a monitorização do ambiente exterior à luminária, o que implicaria uma atualização quer do *firmware* do *gateway* como também da plataforma *web* desenvolvida. Essa atualização tornar-se-á obrigatória para possibilitar que através da plataforma não só se consiga monitorizar dados, como também alterar configurações do sistema.

A parte relativa à comunicação também deveria merecer atenção, uma vez que esta apresenta algumas falhas. O envio quase constante de informação acaba por causar o congestionamento e saturação da rede. A distância entre o gateway e a ponte é outro ponto a considerar uma vez que esta é relativamente elevada.

Bibliografia

[1] MultiSource Technologies: http://ledstreetlight.co.za/
Último acesso: 13/10/2016

[2] Comissão Europeia: http://ec.europa.eu/europe2020/targets/eu-targets/index_pt.htm
Último acesso: 13/10/2016

[3] LITES: http://www.lites-project.eu/pt-pt
Último acesso: 13/10/2016

[4] Comissão Europeia: http://ec.europa.eu/clima/policies/strategies/2020/index_en.htm
Último acesso: 13/10/2016

[5] Câmara Municipal de Lisboa: http://revelarlx.cm-lisboa.pt/gca/?id=1246
Último acesso: 13/10/2016

[6] Henrique Costa. Plataforma Inteligente para Sistemas de Iluminação Pública. Master Thesis. Universidade de Aveiro – Departamento de Eletrónica, Telecomunicações e Informática, 2013

[7] Daniel Ribau Lourenço. Sistemas de Iluminação Pública com Gestão Inteligente de Consumo. Master Thesis. Universidade de Aveiro – Departamento de Eletrónica, Telecomunicações e Informática, 2010

[8] Pedro Fonseca. Sistemas de Instrumentação Electrónica. Universidade de Aveiro, 2011/12

[9] Cypress MicroSystems:“Pyroelectric Infrared Motion Detector, PSoC Style” AN2105, 2003

[10] Tridonic: http://www.corridorfunction.com/corridorFUNCTION/configuration.html
Último acesso: 13/10/2016

[11] Texas Instruments: “Introduction to the Controller Area Network (CAN)”. Application Report. 2008.

[12] Muhammad Salman Yousuf and Mustafa El-Shafei, “Power Line Communications: An Overview – Part I” IEEE, 2008

[13] Dr. Haim Mazar: Short Range Devices (SRDs) and Ultra Wide Band (UWB), ITU Workshop 2014.

[14] Zigbee: http://www.zigbee.org/zigbeealliance/our-members/,
Último acesso: 13/10/2016

[15] EE Times: http://www.eetimes.com/document.asp?doc_id=1274115,
Último acesso: 13/10/2016

[16] Arduino Nano: https://www.arduino.cc/en/Main/ArduinoBoardNano

[17] Nordic Semiconductor nRF24L01+ Single Chip 2.4GHz Transceiver. Preliminary Product Specification v1.0. July 2007.

[18] Mean Well, 40W Single Output LED Driver NPF-40D series.

[19] Tridonic. Talex module STARK LLE 24-280-1250.

[20] AOSONG, Temperature and Humidity module, AM2302 Product Manual.

[21] Itead Studio, Electronic Brickof Electronic Meter Sensor, 2013

[22] Analog Devices, Digital Accelerometer - ADXL345.2016.

[23] Bosch, Data Sheet BMP180 Digital pressure sensor, Bosch Sensortec, 2013

[24] Analog Devices, I2C Compatible, 256-Position Digital Potentiometers, Data Sheet.

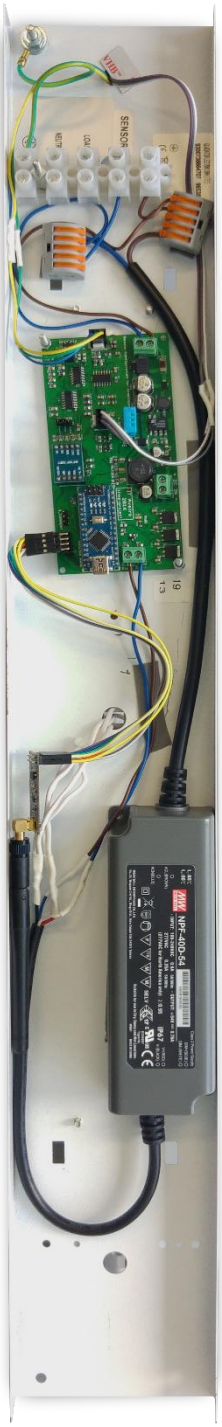
[25] http://sim.okawa-denshi.jp/en/OPseikiLowkeisan.htm
Último acesso: 02/11/2016

[26] BeagleBone Black: http://elinux.org/Beagleboard:BeagleBoneBlack
Último acesso: 23/10/2016

Anexos

Anexo A – Luminária

- **A.1 - Vista interior**



- **A.2 - Vista exterior**

Anexo B – *Hardware*

- **B.1 - Esquemático do Módulo de Controlo**

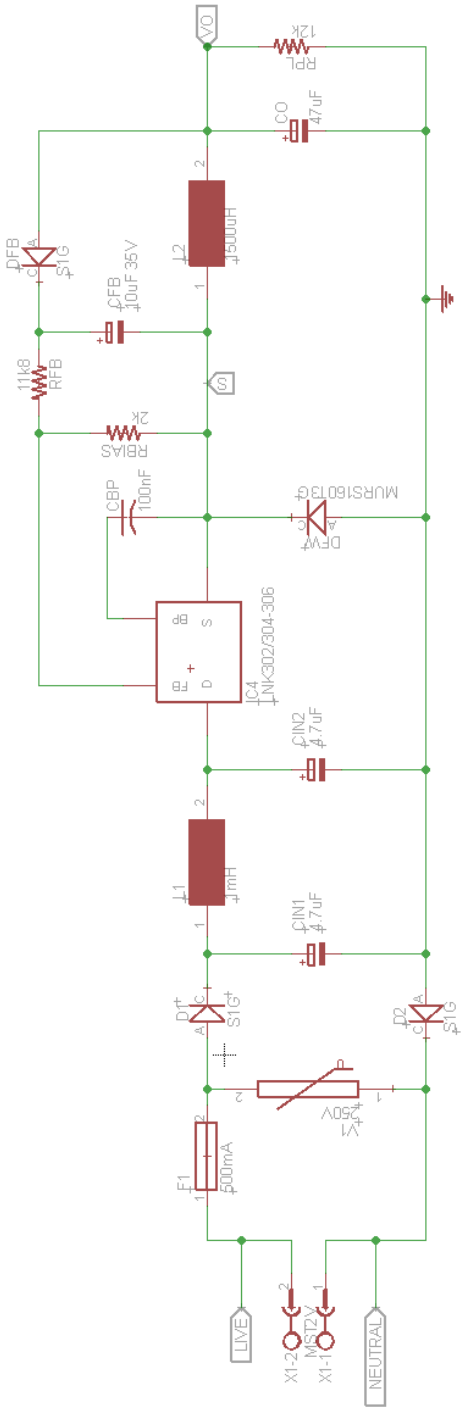


Figura B.1 – Circuito de comutação (conversão CA/CC)

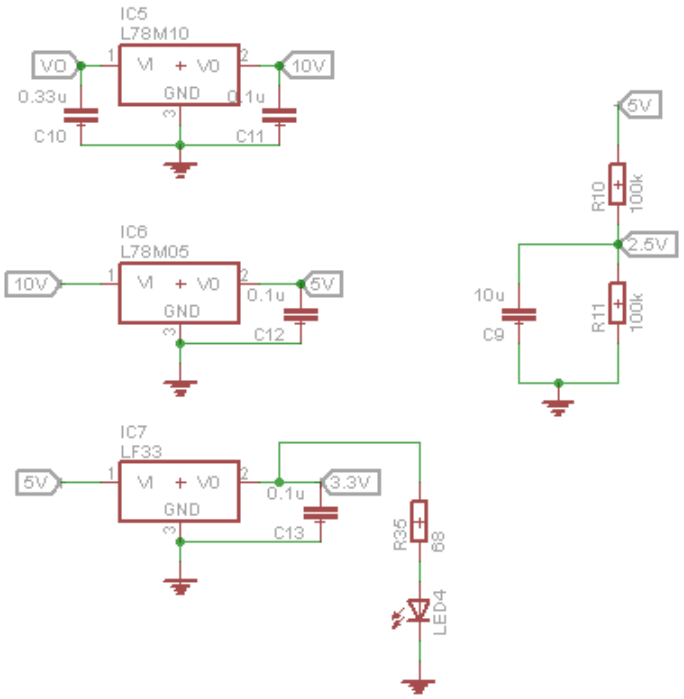


Figura B.2 – Conversão DC/DC

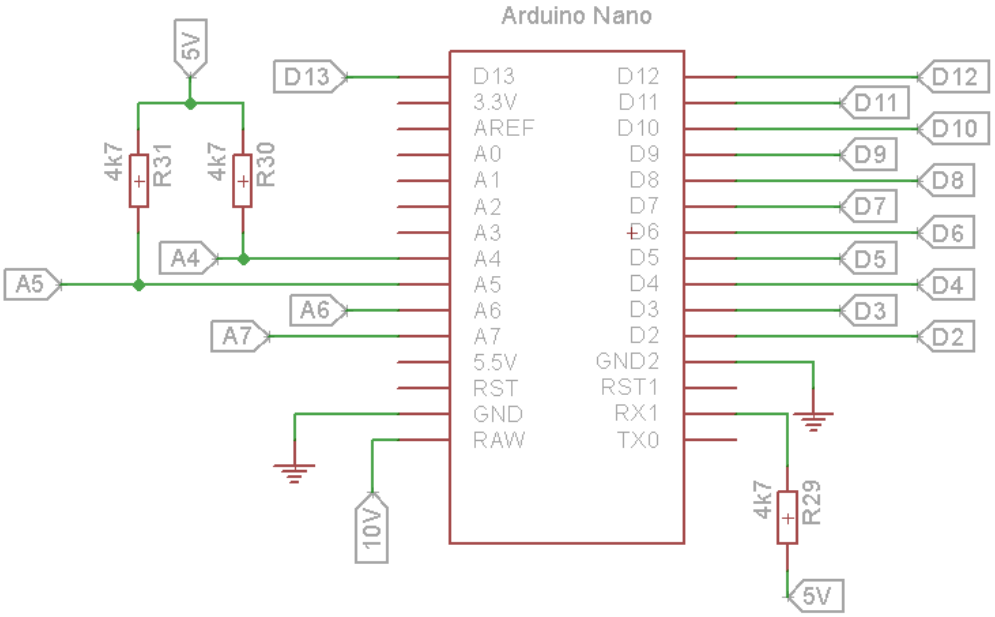


Figura B.3 – Microcontrolador

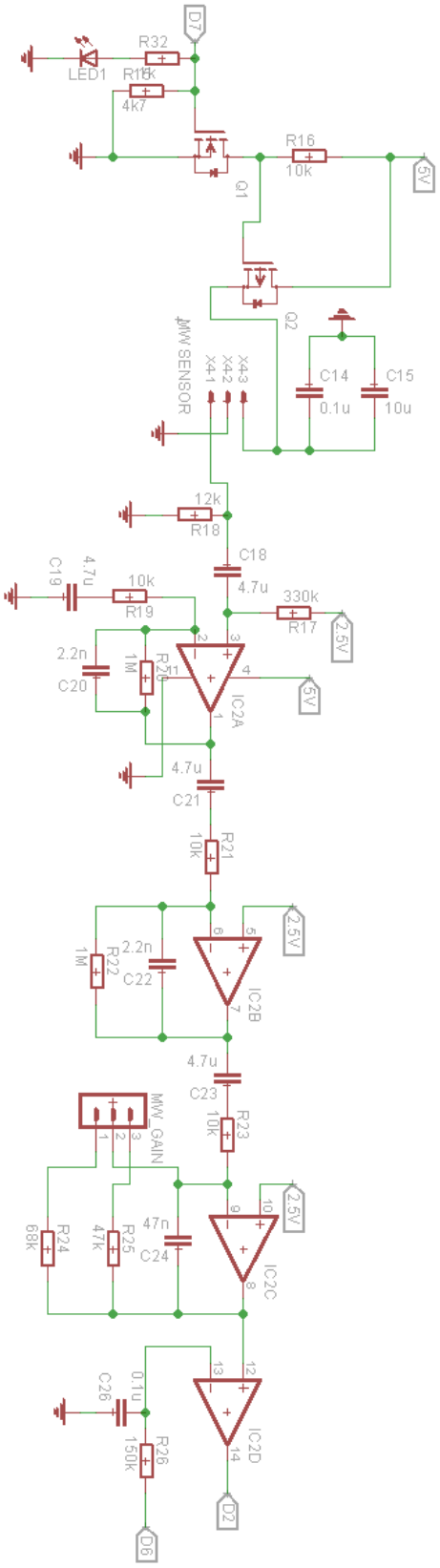


Figura B.4 – Circuito do sensor de micro-ondas

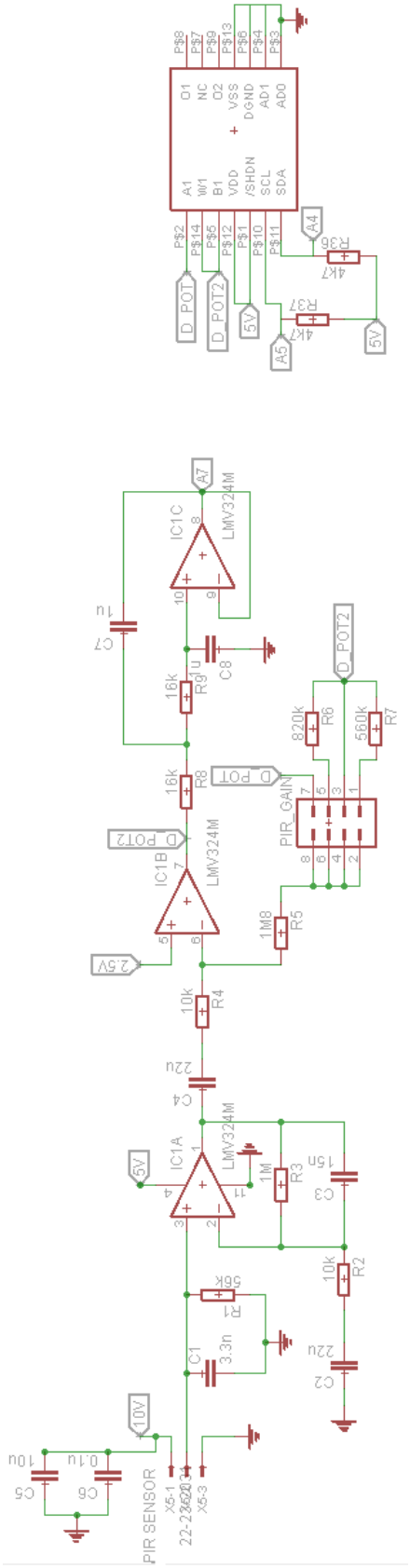


Figura B.5 – Circuito do sensor PIR

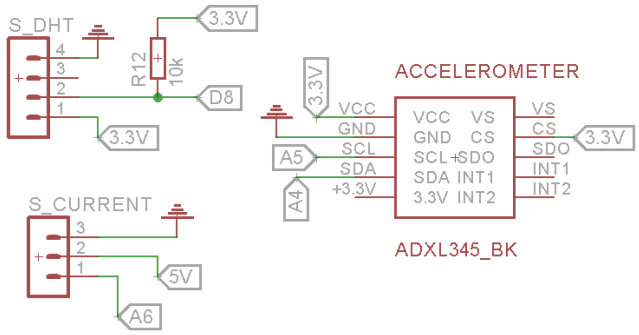


Figura B.6 – Sensores de corrente, temperatura e humidade e acelerómetro.

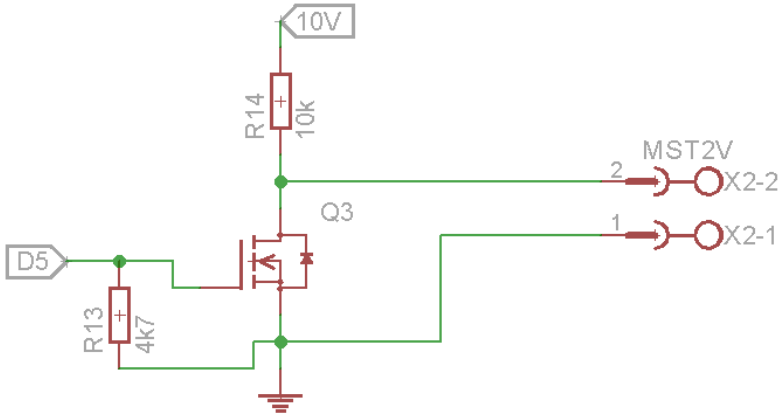


Figura B.7 – Saída de PWM para o LED driver

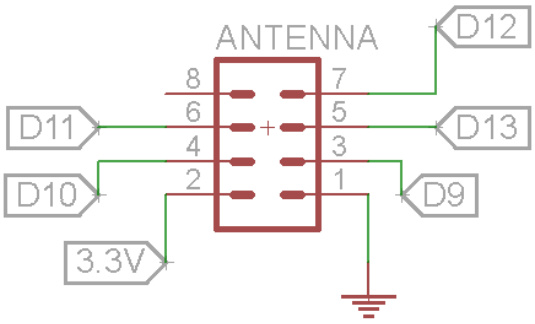


Figura B.8 – Módulo $nRF24L01$

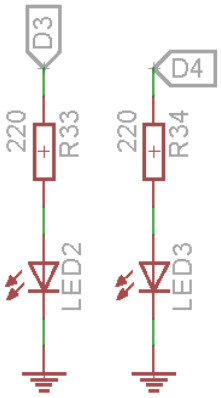
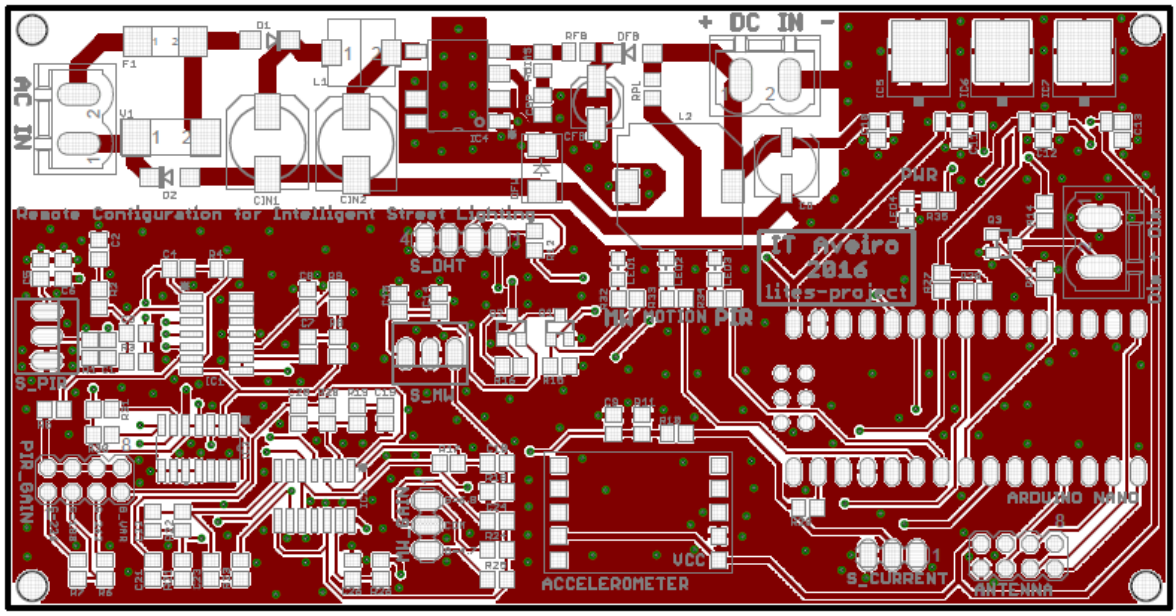


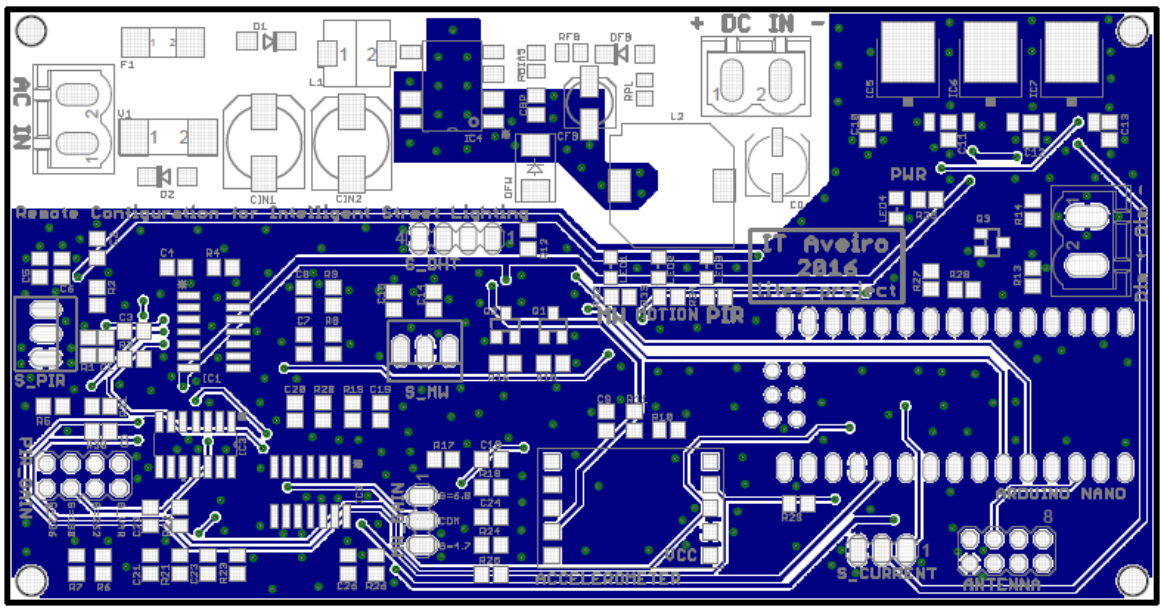
Figura B.9 – *Debug* LEDs

- ***B.2* – PCB do Módulo de Controlo:**

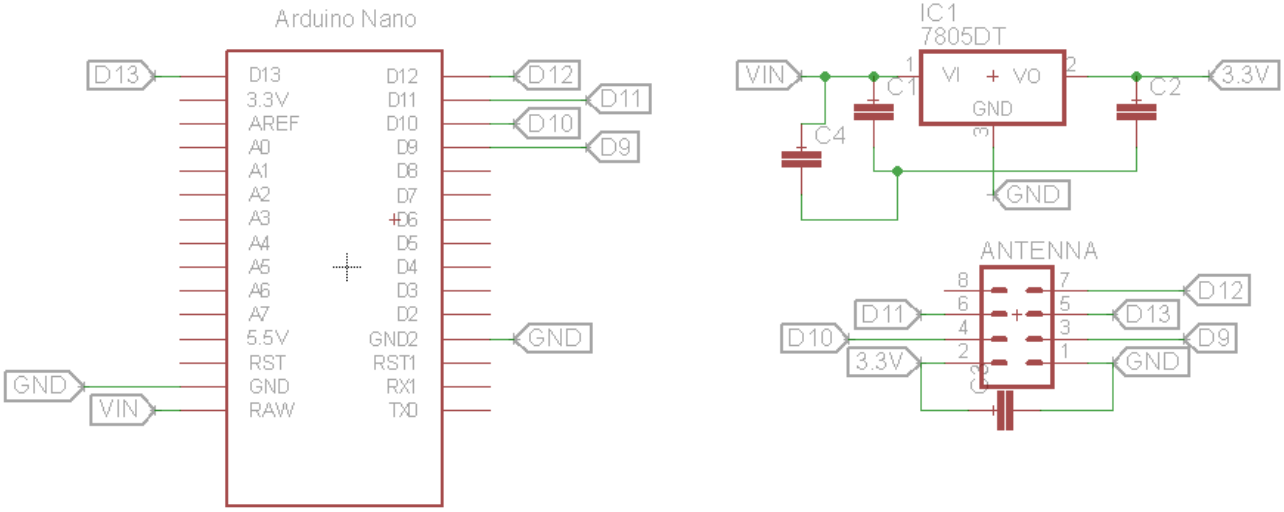
  - *Top layer:*



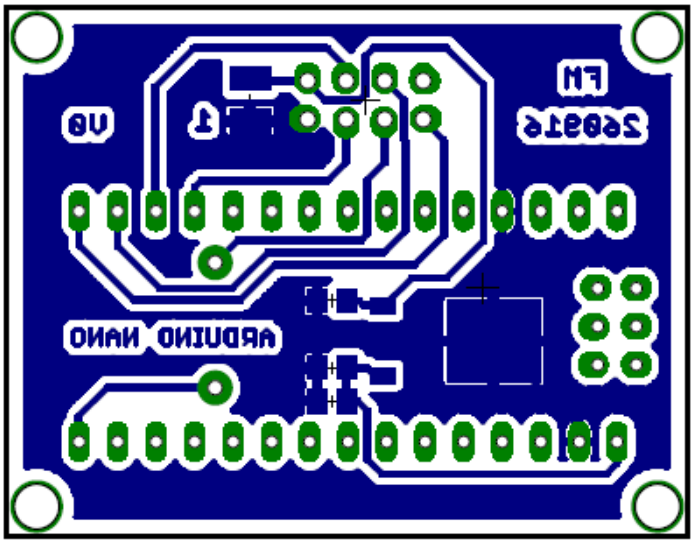
  - *Bottom layer:*



- **B.3 – Esquemático do Módulo de Gestão Remota:**



- **B.4 – PCB do Módulo de Gestão Remota:**



Anexo C – *Firmware* do Módulo de Controlo

```
#include <TimerOne.h>
#include <EEPROM.h>
#include <DHT.h>
#include <ADXL345.h>
#include "printf.h"
#include "nRF24L01.h"
#include "RF24.h"
#include <SPI.h>
#include <Wire.h>

#define TRUE                1
#define FALSE               0

#define MONITORING_MSG      68
#define MOTION_DETECT_MSG   69
#define READ_CONF_MSG       70
#define CONFIGURATION_MSG   71
#define READ_DATA           72

#define RUNNING             1
#define READ_SENSORS        2
#define SEND_DATA           3

#define MW_IN_PIN           2
#define GREEN_LED           3
#define RED_LED             4
#define OUT_PWM_PIN         5
#define MW_SET_THRESH_PIN   6
#define MW_ONOFF_PIN        7
#define DHT_PIN             8

#define CURRENT_SENSOR      A3
#define PIR_SENSOR_IN       A7

#define MAX_LIGHT           0
#define DIM_LIGHT           1
#define MIN_LIGHT           2
#define NO_LIGHT            3

#define PWR_LOW             0
#define PWR_MAX             1

/*-------- NRF24 ----------*/
RF24 radio(9, 10);
byte my_node = 7;
byte gateway_node = 0;
byte config_node = 55;
//                            READ PIPE    WRITE PIPE GATEWAY WRITE PIPE CONFIG
const uint64_t pipes[3] = { 0xF0F0F0D25ALL, 0xF0F0F0F05ALL, 0xABCDABCD71LL};
//Beagle Pipes:
//W: 0xF0F0F0D25A
//R: 0XF0F0F0F05A
//R: 0XF0F0F0F069
//R: 0XF0F0F0F096
//R: 0XF0F0F0F0A5
//R: 0XF0F0F0F0C3
/*--------- ADXL ----------*/
ADXL345 adxl;
uint8_t Th_adxl = 40;
```

```
boolean firstRead = TRUE;
float top;
float bottom;
float mean;
/*------ DHT SENSOR ---------*/
#define DHT_TYPE DHT22
DHT dht(DHT_PIN, DHT_TYPE);
/*---------- MOTION SENSOR -----------*/
uint16_t pir_measures[5];
volatile int counter = 0;
volatile bool ext_sensor_on = FALSE;
/*---------- CURRENT SENSOR -----------*/
#define N_SAMPLES_I   10
volatile uint8_t I_n_sample = 0;
volatile uint16_t I_sample[N_SAMPLES_I];
/*--------- SYSTEM ---------*/
byte SYS_STATE = RUNNING;

uint8_t L_profile = NO_LIGHT;
bool isLightProfileRunning = FALSE;

volatile unsigned long T_ADXL;
volatile unsigned long T_SENSORS;
volatile unsigned long T_PIR;
volatile unsigned long T_LIGHT;
unsigned long T_CHK_MOTION;
unsigned long T_SEND_DATA;

uint8_t data_received[22];
uint8_t data2send[22];
uint8_t garbage[22];

struct payload_t {
  //header
  uint16_t this_node;
  uint16_t other_node;
  uint16_t packet_id;
  uint16_t message_type;
  //data
  byte temperature;
  byte humidity;
  boolean vibration;
  char illuminationState;
  float ef_current;
  int vibration1;
  int vibration2;
  int vibration3;
};

struct config_t {
  //Light profile
  uint8_t maxLight;
  uint8_t minLight;
  uint16_t maxTime;
  uint8_t dimTime;
  uint16_t minTime;

  //Sensors
  uint8_t PIR_gain;
  uint8_t PIR_thresh;

  uint8_t MW_thresh;
  uint8_t MW_Npulses;
```

```
123.
124.   uint8_t ADXL_thresh;
125.   // Data transmission period
126.   uint8_t infoTime;
127. };
128.
129.
130. struct system_t {
131.
132.   float maxLight;
133.   float minLight;
134.   unsigned long int T_lvl_max;
135.   unsigned long int T_lvl_dim;
136.   unsigned long int T_lvl_min;
137.
138.   //Sensors parameters
139.   float PIR_Gain;
140.   float PIR_Thresh;
141.
142.   float MW_Thresh;
143.   uint8_t MW_nPulses;
144.
145.   uint8_t ADXL_Thresh;
146.
147.   unsigned long int Data_T;
148. };
149.
150. payload_t data_payload;
151. config_t luminary_config;
152. system_t sys;
153.
154. void init_parameters(void);
155. void ADXLsensor_read(void);
156. void DHTsensor_read(void);
157. void ADXLsensor_config(int);
158. void handle_Config_msg(void);
159. void handle_MotionDetect_msg(void);
160. void handle_ReadConf_msg(void);
161. void PulseCounter();
162. void PIR_detection(void);
163. void setLightProfile(void);
164. void radio_config(uint8_t);
165.
166. void setup(void)
167. {
168.   Serial.begin(115200);
169.   while (!Serial);
170.
171.   SPI.begin();
172.   Wire.begin();
173.   init_parameters();
174.   dht.begin();
175.   ADXLsensor_config(sys.ADXL_Thresh);
176.   radio.begin();
177.   printf_begin();
178.   radio_config(PWR_MAX);
179.   ports_config();
180.   radio.startListening();
181.
182.   T_PIR = millis();
183.   T_SENSORS = millis();
184.   T_ADXL = millis();
185.   T_CHK_MOTION = millis();
```

```
186.   T_SEND_DATA = millis();
187. }
188.
189. void loop(void) {
190.
191.   LightProfile();
192.
193.   if (radio.available())
194.   {
195.     radio.stopListening();
196.     radio.read(&data_received, sizeof(data_received));
197.
198.     uint8_t data_type = data_received[6]; //Check the received message type
199.
200.     switch (data_type) {
201.
202.       case CONFIGURATION_MSG:
203.         handle_Config_msg();
204.         break;
205.
206.       case READ_CONF_MSG:
207.         handle_ReadConf_msg();
208.         break;
209.
210.       case READ_DATA:
211.         handle_ReadData_msg();
212.         break;
213.
214.       default:
215.         break;
216.     };
217.     radio.startListening();
218.   }
219.
220.   switch (SYS_STATE)
221.   {
222.     case RUNNING:
223.
224.       if (millis() - T_CHK_MOTION > 250)
225.       {
226.         if (counter > sys.MW_nPulses)
227.         {
228.           radio.stopListening();
229.           radio_config(PWR_LOW);
230.           data2send[0] = my_node;
231.           data2send[1] = config_node;
232.           data2send[2] = 0; data2send[3] = 0; data2send[4] = 0; data2send[5] = 0;
233.           data2send[6] = MOTION_DETECT_MSG;
234.           radio.openWritingPipe(pipes[2]);
235.           radio.write(&data2send, sizeof(data2send));
236.
237.           radio.startListening();
238.           isLightProfileRunning = TRUE;
239.           T_LIGHT = millis();
240.           digitalWrite(GREEN_LED, HIGH);
241.
242.         } else
243.           digitalWrite(GREEN_LED, LOW);
244.
245.         T_CHK_MOTION = millis();
246.         counter = 0;
247.       }
```

```
248.      else if ((millis() - T_PIR) > 10 && isLightProfileRunning == FALSE)
249.      {
250.        PIR_detection();
251.        T_PIR = millis();
252.      }
253.      else if ((millis() - T_ADXL) > 5000 && isLightProfileRunning == FALSE)
254.      {
255.        ADXLsensor_read();
256.        T_ADXL = millis();
257.      }
258.      else if (millis() - T_SENSORS > 10000 && isLightProfileRunning == FALSE)
259.      {
260.        T_SENSORS = millis();
261.        SYS_STATE = READ_SENSORS;
262.      }
263.
264.      if (millis() - T_SEND_DATA > sys.Data_T)
265.      {
266.        digitalWrite(GREEN_LED, HIGH);
267.        sendData();
268.        T_SEND_DATA = millis();
269.        digitalWrite(GREEN_LED, LOW);
270.      }
271.
272.      break;
273.
274.    case READ_SENSORS:
275.      radio.stopListening();
276.      DHTsensor_read();
277.      radio.startListening();
278.      SYS_STATE = RUNNING;
279.      break;
280.  }
281.
282.  Serial.flush();
283.}
284.
285.void sendData(void)
286.{
287.  radio.stopListening();
288.  radio_config(PWR_MAX);
289.
290.  data_payload.this_node = my_node;
291.  data_payload.other_node = gateway_node;
292.  data_payload.packet_id += 1;
293.  data_payload.message_type = MONITORING_MSG;
294.
295.  radio.openWritingPipe(pipes[1]);
296.  radio.write(&data_payload, sizeof(data_payload));
297.
298.  data_payload.vibration = FALSE;
299.  data_payload.vibration1 = 0;
300.  data_payload.vibration2 = 0;
301.  data_payload.vibration3 = 0;
302.  data_payload.ef_current = 28;
303.  data_payload.illuminationState = 'O';
304.  radio.startListening();
305.}
306.
307.
308.void handle_ReadData_msg(void)
309.{
310.  radio.stopListening();
```

```
311.   radio_config(PWR_LOW);
312.   Serial.println("Send to Master");
313.
314.   data2send[0] = my_node;
315.   data2send[1] = config_node;
316.   data2send[2] = 0; data2send[3] = 0; data2send[4] = 0; data2send[5] = 0;
317.   data2send[6] = READ_DATA;
318.   data2send[7] = 0;
319.   data2send[8] = data_payload.temperature;
320.   data2send[9] = data_payload.humidity;
321.   data2send[10] = data_payload.vibration;
322.   data2send[11] = data_payload.illuminationState;
323.   uint32_t aux = data_payload.ef_current;
324.   data2send[12] = aux & 0xFF;
325.   data2send[13] = (aux & 0xFF) >> 8;
326.   data2send[14] = 1; data2send[15] = 5;
327.
328.   data2send[16] = (data_payload.vibration1 & 0xFF);
329.   data2send[17] = (data_payload.vibration1 & 0x00FF) >> 8;
330.
331.   data2send[18] = (data_payload.vibration2 & 0xFF);
332.   data2send[19] = (data_payload.vibration2 & 0x00FF) >> 8;
333.
334.   data2send[20] = (data_payload.vibration3 & 0xFF);
335.   data2send[21] = (data_payload.vibration3 & 0x00FF) >> 8;
336.
337.   radio.openWritingPipe(pipes[2]);
338.   radio.write(&data2send, sizeof(data2send));
339.   radio.startListening();
340. }
341.
342. void handle_Config_msg(void)
343. {
344.   int addr = 0;
345.
346.   luminary_config.maxLight = data_received[7];
347.   luminary_config.minLight = data_received[8];
348.   luminary_config.maxTime = (data_received[10] << 8) | data_received[9];
349.   luminary_config.dimTime = data_received[11];
350.   luminary_config.minTime = (data_received[13] << 8) | data_received[12];
351.   luminary_config.PIR_gain = data_received[14];
352.   luminary_config.PIR_thresh = data_received[15];
353.   luminary_config.MW_thresh = data_received[16];
354.   luminary_config.MW_Npulses = data_received[17];
355.   luminary_config.ADXL_thresh = data_received[18];
356.   luminary_config.infoTime = data_received[19];
357.
358.   EEPROM.write(addr, luminary_config.maxLight);
359.   addr += 1;
360.   EEPROM.write(addr, luminary_config.minLight);
361.   addr += 1;
362.   EEPROM.write(addr, luminary_config.maxTime);
363.   addr += 1;
364.   EEPROM.write(addr, luminary_config.dimTime);
365.   addr += 1;
366.   EEPROM.write(addr, luminary_config.minTime);
367.   addr += 1;
368.   EEPROM.write(addr, luminary_config.PIR_gain);
369.   addr += 1;
370.   EEPROM.write(addr, luminary_config.PIR_thresh);
371.   addr += 1;
372.   EEPROM.write(addr, luminary_config.MW_thresh);
373.   addr += 1;
```

```
374.   EEPROM.write(addr, luminary_config.MW_Npulses);
375.   addr += 1;
376.   EEPROM.write(addr, luminary_config.ADXL_thresh);
377.   addr += 1;
378.   EEPROM.write(addr, luminary_config.infoTime);
379.
380.   set_parameters(); // Configure Luminaire
381.
382.   // Visual feedback to indicate new config
383.   analogWrite(OUT_PWM_PIN, 0);
384.   delay(500);
385.   analogWrite(OUT_PWM_PIN, 255);
386.   delay(500);
387.   analogWrite(OUT_PWM_PIN, 0);
388.   delay(500);
389.   analogWrite(OUT_PWM_PIN, 255);
390.   delay(500);
391.   analogWrite(OUT_PWM_PIN, 0);
392.   delay(500);
393.   analogWrite(OUT_PWM_PIN, 255);
394. }
395.
396. void handle_MotionDetect_msg(void)
397. {
398.   //Motion detected near this luminaire,
399.   //flag activation to turn on MW sensor
400.   ext_sensor_on = TRUE;
401.   radio.read(&garbage, sizeof(garbage));
402. }
403.
404. void handle_ReadConf_msg(void)
405. {
406.   uint8_t addr = 0;
407.
408.   radio.stopListening();
409.   radio_config(PWR_LOW);
410.   data2send[0] = my_node;
411.   data2send[1] = config_node;
412.   data2send[2] = 0; data2send[3] = 0; data2send[4] = 0; data2send[5] = 0;
413.   data2send[6] = READ_CONF_MSG;
414.
415.   // Get last configuration
416.   luminary_config.maxLight = EEPROM.read(addr);
417.   addr += 1;
418.   luminary_config.minLight = EEPROM.read(addr);
419.   addr += 1;
420.   luminary_config.maxTime = EEPROM.read(addr);
421.   addr += 1;
422.   luminary_config.dimTime = EEPROM.read(addr);
423.   addr += 1;
424.   luminary_config.minTime = EEPROM.read(addr);
425.   addr += 1;
426.   luminary_config.PIR_gain = EEPROM.read(addr);
427.   addr += 1;
428.   luminary_config.PIR_thresh = EEPROM.read(addr);
429.   addr += 1;
430.   luminary_config.MW_thresh = EEPROM.read(addr);
431.   addr += 1;
432.   luminary_config.MW_Npulses = EEPROM.read(addr);
433.   addr += 1;
434.   luminary_config.ADXL_thresh = EEPROM.read(addr);
435.   addr += 1;
436.   luminary_config.infoTime = EEPROM.read(addr);
```

```
437.
438.   data2send[7] = luminary_config.maxLight;
439.   data2send[8] = luminary_config.minLight;
440.   data2send[9] = (luminary_config.maxTime & 0x00FF);
441.   data2send[10] = (luminary_config.maxTime & 0x00FF) >> 8;
442.   data2send[11] = luminary_config.dimTime;
443.   data2send[12] = (luminary_config.maxTime & 0x00FF);
444.   data2send[13] = (luminary_config.maxTime & 0x00FF) >> 8;
445.   data2send[14] = luminary_config.PIR_gain;
446.   data2send[15] = luminary_config.PIR_thresh;
447.   data2send[16] = luminary_config.MW_thresh;
448.   data2send[17] = luminary_config.MW_Npulses;
449.   data2send[18] = luminary_config.ADXL_thresh;
450.   data2send[19] = luminary_config.infoTime;
451.
452.   radio.openWritingPipe(pipes[2]);
453.   radio.write(&data2send, sizeof(data2send));
454.   radio.startListening();
455. }
456.
457. void LightProfile(void)
458. {
459.   static uint8_t LightMax, LightMin;
460.   float aux;
461.   static float Light, LightStep;
462.
463.   aux = 255 * sys.maxLight;
464.   LightMax = (uint8_t)(255 - aux);
465.   aux = 255 * sys.minLight;
466.   LightMin = (uint8_t)(255 - aux);
467.
468.   switch (L_profile)
469.   {
470.     case MAX_LIGHT:
471.
472.       data_payload.ef_current = 132;         // Max consumption
473.       data_payload.illuminationState = 'M'; // Luminaire State: Max
474.       Light = LightMax;
475.       counter = 0;
476.       analogWrite(OUT_PWM_PIN, Light);
477.       if (millis() - T_LIGHT > sys.T_lvl_max) {
478.         L_profile = DIM_LIGHT;
479.         LightStep = LightMax - LightMin;
480.         LightStep = 255 - LightStep;
481.         LightStep /= sys.T_lvl_dim;
482.         data_payload.ef_current = 46;     // Mean consumption
483.         data_payload.illuminationState = 'd'; //Luminaire State: dimming
484.       } else if (counter >= sys.MW_nPulses)
485.         T_LIGHT = millis();
486.       break;
487.
488.     case DIM_LIGHT:
489.       while (Light <= LightMin)
490.       {
491.         Light += LightStep;
492.         analogWrite(OUT_PWM_PIN, (int)Light);
493.         delay(1);
494.       }
495.       L_profile = MIN_LIGHT;
496.       T_LIGHT = millis();
497.       Light = LightMin;
498.       break;
499.
```

```
500.     case MIN_LIGHT:
501.
502.       if (sys.T_lvl_min != 0)
503.       {
504.         data_payload.ef_current = 40; //Minimum Consumption
505.         data_payload.illuminationState = 'm'; //Luminaire State: min
506.         analogWrite(OUT_PWM_PIN, (int)Light);
507.
508.         if (millis() - T_LIGHT > sys.T_lvl_min) {
509.           L_profile = NO_LIGHT;
510.         }
511.         else if (counter >= sys.MW_nPulses)
512.         {
513.           L_profile = MAX_LIGHT;
514.           T_LIGHT = millis();
515.         }
516.       } else {
517.         data_payload.ef_current = 40; //Minimum Consumption
518.         data_payload.illuminationState = 'm';//Luminaire State: min
519.         Light = LightMin;
520.         analogWrite(OUT_PWM_PIN, (int)Light);
521.         isLightProfileRunning = FALSE;
522.         if (counter >= sys.MW_nPulses)
523.         {
524.           L_profile = MAX_LIGHT;
525.           T_LIGHT = millis();
526.         }
527.       }
528.       break;
529.
530.     case NO_LIGHT:
531.       analogWrite(OUT_PWM_PIN, 255);
532.       data_payload.ef_current = 28;          //Board Consumption
533.       data_payload.illuminationState = 'O';  //Luminaire State: Off
534.       isLightProfileRunning = FALSE;
535.       if (counter >= sys.MW_nPulses)
536.       {
537.         L_profile = MAX_LIGHT;
538.         T_LIGHT = millis();
539.       }
540.       break;
541.
542.     default:
543.       break;
544.   };
545. }
546./********* ADXL SENSOR FUNCTIONS ********************/
547.void ADXLsensor_read()
548.{
549.   int x, y, z;
550.   float Sqr;
551.   float rms;
552.
553.   if (firstRead)
554.   {
555.     unsigned long st = millis();
556.     while (millis() - st < 500)
557.     {
558.       adxl.readAccel(&x, &y, &z);
559.       Sqr = float((sq(x) + sq(y) + sq(z)) / 3);
560.       float rms = sqrt(Sqr);
561.       mean = (rms + mean) / 2;
562.     }
```

```
    top = mean * 1.05;
    bottom = mean * 0.95;
    firstRead = FALSE;
  }

  adxl.readAccel(&x, &y, &z);
  Sqr = float((sq(x) + sq(y) + sq(z)) / 3);
  rms = sqrt(Sqr);

  if (rms < bottom || rms > top)
  {
    int vibration_value = abs(rms - mean);

    if (vibration_value > data_payload.vibration1) {
      data_payload.vibration1 = vibration_value;
      data_payload.vibration = TRUE;
    }
    else if (vibration_value > data_payload.vibration2) {
      data_payload.vibration2 = vibration_value;
      data_payload.vibration = TRUE;
    }
    else if (vibration_value > data_payload.vibration3) {
      data_payload.vibration3 = vibration_value;
      data_payload.vibration = TRUE;
    }
  }
}

void ADXLsensor_config(int Th)
{
  adxl.powerOn();
  adxl.setActivityThreshold(Th);
  adxl.setInactivityThreshold(Th);
  adxl.setTimeInactivity(10);
  adxl.setActivityX(1);
  adxl.setActivityY(1);
  adxl.setActivityZ(1);
  adxl.setInactivityX(1);
  adxl.setInactivityY(1);
  adxl.setInactivityZ(1);
  adxl.setTapDetectionOnX(1);
  adxl.setTapDetectionOnY(1);
  adxl.setTapDetectionOnZ(1);
  adxl.setInterruptMapping( ADXL345_INT_SINGLE_TAP_BIT, ADXL345_INT1_PIN );
  adxl.setInterruptMapping( ADXL345_INT_DOUBLE_TAP_BIT, ADXL345_INT1_PIN );
  adxl.setInterruptMapping( ADXL345_INT_FREE_FALL_BIT, ADXL345_INT1_PIN );
  adxl.setInterruptMapping( ADXL345_INT_ACTIVITY_BIT, ADXL345_INT1_PIN );
  adxl.setInterruptMapping( ADXL345_INT_INACTIVITY_BIT, ADXL345_INT1_PIN );
  adxl.setInterrupt( ADXL345_INT_SINGLE_TAP_BIT, 0);
  adxl.setInterrupt( ADXL345_INT_DOUBLE_TAP_BIT, 0);
  adxl.setInterrupt( ADXL345_INT_FREE_FALL_BIT, 0);
  adxl.setInterrupt( ADXL345_INT_ACTIVITY_BIT, 1);
  adxl.setInterrupt( ADXL345_INT_INACTIVITY_BIT, 1);
}
/************** DHT SENSOR FUNCTIONS*****************************************/
void DHTsensor_read(void)
{
  data_payload.humidity = dht.readHumidity();
  data_payload.temperature = dht.readTemperature();
}

void PulseCounter()
{
```

```
626.   counter++;
627. }
628.
629. void PIR_detection(void)
630. {
631.   static uint8_t n_sample;
632.   float aux, average = 0.0;
633.   static unsigned long T_MW;
634.   static bool MW_sensor_on = FALSE;
635.
636.   if (n_sample <= 5)
637.   {
638.     pir_measures[n_sample] = analogRead(PIR_SENSOR_IN);
639.     n_sample++;
640.   }
641.   else {
642.
643.     for (int i = 0; i < 5; i++)
644.     {
645.       aux = pir_measures[i] / 5.0;
646.       average += aux;
647.     }
648.
649.     int pirH = sys.PIR_Thresh * 1023;
650.     int pirL = (1 - sys.PIR_Thresh) * 1023;
651.
652.     if (average >= pirH || average <= pirL || ext_sensor_on == TRUE)
653.     {
654.       digitalWrite(RED_LED, HIGH);
655.       digitalWrite(MW_ONOFF_PIN, HIGH);
656.       attachInterrupt(digitalPinToInterrupt(MW_IN_PIN), PulseCounter, FALLING);
657.       MW_sensor_on = TRUE;
658.       ext_sensor_on = FALSE;
659.       T_MW = millis();
660.     } else {
661.       digitalWrite(RED_LED, LOW);
662.
663.       if (MW_sensor_on == TRUE)
664.         if (millis() - T_MW > 5000)
665.         {
666.           MW_sensor_on = FALSE;
667.           digitalWrite(MW_ONOFF_PIN, LOW);
668.           detachInterrupt(digitalPinToInterrupt(MW_IN_PIN));
669.           T_MW = millis();
670.         }
671.     }
672.     average = 0.0;
673.     n_sample = 0;
674.   }
675. }
676.
677. void init_parameters(void)
678. {
679.   uint8_t addr = 0;
680.
681.   sys.maxLight = EEPROM.read(addr) / 100.0;
682.   addr += 1;
683.   sys.minLight = EEPROM.read(addr) / 100.0;
684.   addr += 1;
685.   sys.T_lvl_max = EEPROM.read(addr) * 1000;
686.   addr += 1;
687.   sys.T_lvl_dim = EEPROM.read(addr) * 1000;
688.   addr += 1;
```

```
689.   sys.T_lvl_min = EEPROM.read(addr) * 1000;
690.   addr += 1;
691.   sys.PIR_Gain = EEPROM.read(addr) / 100.0;
692.   addr += 1;
693.   sys.PIR_Thresh = EEPROM.read(addr) / 100.0;
694.   addr += 1;
695.   sys.MW_Thresh = EEPROM.read(addr) / 100.0;
696.   addr += 1;
697.   sys.MW_nPulses = EEPROM.read(addr);
698.   addr += 1;
699.   sys.ADXL_Thresh = EEPROM.read(addr) / 100.0;
700.   addr += 1;
701.   sys.Data_T = EEPROM.read(addr) * 1000;
702.
703.   // EEPROM consistency check
704.   if (sys.maxLight == 0.0 || sys.maxLight > 1.0)
705.     sys.maxLight = 1.0;
706.
707.   if (sys.minLight == 0.0 || sys.maxLight > 1.0)
708.     sys.maxLight = 0.1;
709.
710.   if (sys.maxLight == 0.0 || sys.maxLight > 1.0)
711.     sys.maxLight = 1.0;
712.
713.   if (sys.T_lvl_max == 0 || sys.T_lvl_max > 1800000)
714.     sys.T_lvl_max = 5000;
715.
716.   if (sys.T_lvl_dim == 0 || sys.T_lvl_dim > 120000)
717.     sys.T_lvl_dim = 2000;
718.
719.   if (sys.T_lvl_min > 180000)
720.     sys.T_lvl_min = 5000;
721.
722.   if (sys.PIR_Gain == 0.0 || sys.PIR_Gain > 1.0)
723.     sys.PIR_Gain = 0.9;
724.
725.   if (sys.PIR_Thresh < 0.6 || sys.PIR_Thresh > 1.0)
726.     sys.PIR_Thresh = 0.6;
727.
728.   if (sys.MW_Thresh < 0.6 || sys.MW_Thresh > 1.0)
729.     sys.MW_Thresh = 0.6;
730.
731.   if (sys.MW_nPulses == 0 || sys.MW_nPulses > 20)
732.     sys.MW_nPulses = 3;
733.
734.   if (sys.ADXL_Thresh == 0 ||  sys.ADXL_Thresh > 100)
735.     sys.ADXL_Thresh = 40;
736.
737.   if (sys.Data_T < 1000 ||  sys.Data_T > 200000)
738.     sys.Data_T = 10000;
739.
740. }
741.
742. void set_parameters(void)
743. {
744.   sys.maxLight = (luminary_config.maxLight / 100.0);
745.   sys.minLight = (luminary_config.minLight / 100.0);
746.
747.   sys.T_lvl_max = luminary_config.maxTime * 1000;
748.   sys.T_lvl_dim = luminary_config.dimTime * 1000;
749.   sys.T_lvl_min = luminary_config.minTime * 1000;
750.
751.   sys.PIR_Gain = (luminary_config.PIR_gain / 100.0);
```

```
  Wire.beginTransmission(44);
  Wire.write(0x00);
  uint8_t val;
  val = (uint8_t)(sys.PIR_Gain * 255);
  Wire.write(val);
  Wire.endTransmission();

  sys.PIR_Thresh = (luminary_config.PIR_thresh / 100.0);

  sys.MW_Thresh = (luminary_config.MW_thresh / 100.0);
  analogWrite(MW_SET_THRESH_PIN, int(sys.MW_Thresh * 255));
  sys.MW_nPulses = luminary_config.MW_Npulses;

  firstRead = TRUE;
  sys.ADXL_Thresh = luminary_config.ADXL_thresh;
  ADXLsensor_config(sys.ADXL_Thresh);

  sys.Data_T = luminary_config.infoTime * 1000;
}

void radio_config(uint8_t power)
{
  radio.setChannel(0x60);
  radio.setRetries(15, 15);

  if (power == PWR_LOW)
    radio.setPALevel(RF24_PA_LOW);
  else if (power == PWR_MAX)
    radio.setPALevel(RF24_PA_MAX);

  radio.setDataRate(RF24_250KBPS);
  radio.setPayloadSize(22);
  radio.setAutoAck(1);
  radio.setCRCLength(RF24_CRC_16);
  radio.openWritingPipe(pipes[1]);
  radio.openReadingPipe(1, pipes[0]);
  radio.printDetails();
}

void ports_config(void)
{
  pinMode(CURRENT_SENSOR, INPUT);

  pinMode(OUT_PWM_PIN, OUTPUT);
  pinMode(GREEN_LED, OUTPUT);
  pinMode(RED_LED, OUTPUT);
  pinMode(MW_ONOFF_PIN, OUTPUT);

  digitalWrite(OUT_PWM_PIN, HIGH);
  digitalWrite(GREEN_LED, LOW);
  digitalWrite(RED_LED, LOW);
  digitalWrite(MW_ONOFF_PIN, LOW);

  pinMode(MW_IN_PIN, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(MW_IN_PIN), PulseCounter, FALLING);
  detachInterrupt(digitalPinToInterrupt(MW_IN_PIN));
  analogWrite(MW_SET_THRESH_PIN, int(255 * sys.MW_Thresh));
}
//**************** Current Measure Functions ***************************
void readCurrent(void)
{
  float amp_current;
  uint16_t max_value;
```

```
815.
816.   max_value = get_iMaxValue();
817.   amp_current = (float)(max_value / 1024.0 * 1.1 / 200.0 * 1000000);
818.
819.   data_payload.ef_current = amp_current / 1.414;
820. }
821.
822. uint16_t get_iMaxValue(void)
823. {
824.   uint16_t maxVal = 0;
825.
826.   for (int i = 0; i < I_n_sample; i++)
827.   {
828.     if (maxVal < I_sample[i])
829.       maxVal = I_sample[i];
830.
831.     I_sample[i] = 0;
832.   }
833.
834.   I_n_sample = 0;
835.   return maxVal;
836. }
837.
838. void get_Isample(void)
839. {
840.   if (I_n_sample < 10)
841.   {
842.     I_sample[I_n_sample] = analogRead(CURRENT_SENSOR);
843.     I_n_sample++;
844.   } else
845.     Timer1.detachInterrupt();
846. }
```

Anexo D – *Firmware* do Módulo de Gestão Remota

```
#include "nRF24L01.h"
#include "RF24.h"
#include "printf.h"
#include <SPI.h>

#define TRUE                1
#define FALSE               0

#define MONITORING_MSG      68
#define MOTION_DETECT_MSG   69
#define READ_CONF_MSG       70
#define CONFIGURATION_MSG   71
#define READ_DATA           72

RF24 radio(9, 10);

const char my_node = 55;
const char other_node = 7;
uint16_t destination_node;
//                            READ PIPE          WRITE PIPE
const uint64_t pipes[2] = {0xABCDABCD71LL,  0xF0F0F0D25ALL};

bool isConfDone = TRUE;

struct data_t {
  byte temperature;
  byte humidity;
  boolean vibration;
  char illuminationState;
  float ef_current;
  int vibration1;
  int vibration2;
  int vibration3;
};

struct config_t {

  uint8_t maxLight;
  uint8_t minLight;
  uint16_t maxTime;
  uint8_t dimTime;
  uint16_t minTime;

  uint8_t PIR_gain;
  uint8_t PIR_thresh;

  uint8_t MW_thresh;
  uint8_t MW_Npulses;

  uint8_t ADXL_thresh;

  uint8_t infoTime;
};

data_t luminary_data;
config_t luminary_config;

uint8_t data_received[22];
uint8_t data2send[22];
```

```
uint8_t garbage[22];

void radio_config(void);

void setup(void)
{
  Serial.begin(115200);
  while (!Serial);

  SPI.begin();
  radio.begin();
  printf_begin();
  radio_config();
  radio.startListening();
  command_list_menu();
}

void loop() {

  if (isConfDone == FALSE)
  {
    isConfDone = menu();
    command_list_menu();
  }

  if (Serial.available()) {
    char cmd = toupper(Serial.read());

    if (cmd == 'T') {
      radio.stopListening();
      destination_node = ChooseLuminaire();
      Serial.print("\nSending Configuration...");
      packet_creator(destination_node, CONFIGURATION_MSG);
      radio.write(&data2send, sizeof(data2send));
      radio.startListening();
    }
    else if (cmd == 'R') {
      radio.stopListening();
      destination_node = ChooseLuminaire();
      Serial.println("Reading Parameters...");
      packet_creator(destination_node, READ_CONF_MSG);
      radio.write(&data2send, sizeof(data2send));
      radio.startListening();
    }
    else if (cmd == 'D') {
      radio.stopListening();
      Serial.println("Reading Data...");
      packet_creator(destination_node, READ_DATA);
      radio.write(&data2send, sizeof(data2send));
      radio.startListening();
    }
    else if (cmd == 'S') {
      isConfDone = FALSE;
    }
    else {
      Serial.println("ERROR: Unknown command...");
      isConfDone = FALSE;
    }
  }
  else {
    if (radio.available()) {

      radio.stopListening();
```

```
      radio.read(&data_received, sizeof(data_received));
      byte data_type = data_received[6];

      switch (data_type) {

        case READ_DATA:
          handle_ReadData_msg();
          command_list_menu();
          break;

        case MOTION_DETECT_MSG:
          handle_MotionDetect_msg();
          command_list_menu();
          break;

        case READ_CONF_MSG:
          handle_ReadConf_msg();
          command_list_menu();
          break;

        default:
          Serial.print("Unknown Data Received:");
          radio.read(&garbage, sizeof(garbage));
          break;
      };
    }
  }
}

uint8_t ChooseLuminaire(void)
{
  uint8_t aux, addr;

Insert_Lnumber:
  Serial.print("Send to Luminaire #: ");

  while (!Serial.available());

  aux = Serial.parseInt();

  if (aux == 11) {
    Serial.print(aux);
    addr = 7;
  } else {
    Serial.println("Unknown Luminaire");
    goto Insert_Lnumber;
  }
  return addr;
}

bool menu(void)
{
  char incomingByte;
  uint16_t L_integer = 0;

  Serial.println("************* Configuration Menu *************");
  Serial.println("----------------------------------------------");
  Serial.println("--------- Light Profile Configuration --------");

  /******************* maxLight ********************************/
Insert_maxLight:
  Serial.print("MaxLight(0-100%): ");

```

```
  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 0 || L_integer > 100) {
    Serial.println("Invalid:(0-100%)");
    goto Insert_maxLight;
  }
  luminary_config.maxLight = L_integer;
  Serial.println(luminary_config.maxLight);
  /******************** minLight ********************************/
Insert_minLight:
  Serial.print("MinLight(0-100%): ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 0 || L_integer > 100) {
    Serial.println("Invalid:(0-100%)");
    goto Insert_minLight;
  }
  luminary_config.minLight = L_integer;
  Serial.println(luminary_config.minLight);
  /******************** Time on Max ********************************/
Insert_maxTime:
  Serial.print("MaxTime(sec): ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 1 || L_integer > 1800) {
    Serial.println("Invalid: (1 - 1800 sec)");
    goto Insert_maxTime;
  }
  luminary_config.maxTime = L_integer;
  Serial.println(luminary_config.maxTime);
  /******************** Time on Dim ********************************/
Insert_dimTime:
  Serial.print("DimmTime(sec): ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 1 || L_integer > 120) {
    Serial.println("Invalid: (1 - 120 sec)");
    goto Insert_dimTime;
  }
  luminary_config.dimTime = L_integer;
  Serial.println(luminary_config.dimTime);
  /******************** Time on Min ********************************/
Insert_minTime:
  Serial.print("MinTime[0 -> always on](sec): ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 0 || L_integer > 1800) {
    Serial.println("Invalid:(0 - 1800 sec)");
    goto Insert_minTime;
```

```
  }
  luminary_config.minTime = L_integer;
  Serial.println(luminary_config.minTime);
  Serial.println("-----------------------------------------------");
  Serial.println("------------- Sensors Configuration ----------");
  /******************** R value *********************************/
Insert_PIR_gain:
  Serial.println("PIR: ");
  Serial.print("Gain(1-100%): ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 1 || L_integer > 100) {
    Serial.println("Invalid:(1-100%)");
    goto Insert_PIR_gain;
  }
  luminary_config.PIR_gain = L_integer;
  Serial.println(luminary_config.PIR_gain);
  /******************** PIR Thresh *********************************/
Insert_PIR_thresh:
  Serial.print("Thresh(60-100%): ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 60 || L_integer > 100) {
    Serial.println("Invalid:(60-100%)");
    goto Insert_PIR_thresh;
  }
  luminary_config.PIR_thresh = L_integer;
  Serial.println(luminary_config.PIR_thresh);
  /******************** MW Thresh *********************************/
Insert_MW_thresh:
  Serial.println("MW: ");
  Serial.print("Thresh(60-100%): ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 60 || L_integer > 100) {
    Serial.println("Invalid:(60-100%)");
    goto Insert_MW_thresh;
  }
  luminary_config.MW_thresh = L_integer;
  Serial.println(luminary_config.MW_thresh);
  /******************** MW nPulses *********************************/
Insert_MW_Npulses:
  Serial.print("Npulses: ");

  while (!Serial.available());

  L_integer = Serial.parseInt();

  if (L_integer < 1 || L_integer > 20) {
    Serial.println("Invalid: (1-20 pulses)");
    goto Insert_MW_Npulses;
  }
  luminary_config.MW_Npulses = L_integer;
  Serial.println(luminary_config.MW_Npulses);
```

```
312.   /******************* ADLX Threshold ********************************/
313. Insert_ADXL_thresh:
314.   Serial.println("ADXL: ");
315.   Serial.print("Thresh: ");
316.   while (!Serial.available());
317.
318.   L_integer = Serial.parseInt();
319.
320.   if (L_integer < 1 || L_integer > 100) {
321.     Serial.println("Value:(1-100)");
322.     goto Insert_ADXL_thresh;
323.   }
324.   luminary_config.ADXL_thresh = L_integer;
325.   Serial.println(luminary_config.ADXL_thresh);
326.   /******************* Infor Period ********************************/
327. Insert_infoTime:
328.   Serial.print("Data acquisition(sec): ");
329.
330.   while (!Serial.available());
331.
332.   L_integer = Serial.parseInt();
333.
334.   if (L_integer < 1 || L_integer > 200) {
335.     Serial.println("Invalid:(1-200)");
336.     goto Insert_infoTime;
337.   }
338.   luminary_config.infoTime = L_integer;
339.   Serial.println(luminary_config.infoTime);
340.   Serial.println("-------------- END CONFIGURATION ------------");
341.
342.   return TRUE;
343. }
344.
345. void handle_MotionDetect_msg(void)
346. {
347.   Serial.println("\nMOTION DETECTED !!");
348.   radio.read(&garbage, sizeof(garbage));
349. }
350.
351. void handle_ReadConf_msg(void)
352. {
353.   uint16_t aux_u;
354.
355.   Serial.println("----------------------------------");
356.   Serial.print("Luminary "); Serial.print(data_received[0]); Serial.println(" con
    figured parameters:");
357.   Serial.print("MaxLight(%): ");
358.   Serial.println(data_received[7]);
359.   Serial.print("MinLight(%): ");
360.   Serial.println(data_received[8]);
361.   Serial.print("MaxTime(sec): ");
362.   aux_u = 0;
363.   aux_u = (data_received[10] << 8) | data_received[9];
364.   Serial.println(aux_u);
365.   Serial.print("DimmTime(sec): ");
366.   Serial.println(data_received[11]);
367.   Serial.print("MinTime(sec): ");
368.   aux_u = 0;
369.   aux_u = (data_received[13] << 8) | data_received[12];
370.   Serial.println(aux_u);
371.   Serial.print("PIR Gain(%): ");
372.   Serial.println(data_received[14]);
373.   Serial.print("PIR Thresh(%): ");
```

```
374.   Serial.println(data_received[15]);
375.   Serial.print("MW Thresh(%): ");
376.   Serial.println(data_received[16]);
377.   Serial.print("MW nPulses: ");
378.   Serial.println(data_received[17]);
379.   Serial.print("ADXL Thresh: ");
380.   Serial.println(data_received[18]);
381.   Serial.print("Data acquisition(sec): ");
382.   Serial.println(data_received[19]);
383.   Serial.println("----------------------------------");
384. }
385.
386. void handle_ReadData_msg(void)
387. {
388.   uint32_t aux_f = 0;
389.   uint16_t aux_u = 0;
390.
391.   Serial.print("Luminary "); Serial.print(data_received[0]); Serial.println(" val
    ues:");
392.   Serial.print("Temperature: ");
393.   Serial.println(data_received[8]);
394.   Serial.print("Humidity: ");
395.   Serial.println(data_received[9]);
396.   Serial.print("Vibration:");
397.   Serial.println(data_received[10]);
398.   Serial.print("Illumination State:");
399.   Serial.write(data_received[11]);
400.   Serial.print("\nCurrent:");
401.   aux_f = (data_received[15] << 24) | data_received[14] << 16;
402.   aux_f = aux_f | (data_received[13] << 8);
403.   aux_f = aux_f | data_received[12];
404.   Serial.println(aux_f);
405.   Serial.println("Vibration Values:");
406.   Serial.print("Vibration 1: ");
407.   aux_u = (data_received[17] << 8) | data_received[16];
408.   Serial.println(aux_u);
409.   aux_u = 0;
410.   aux_u = (data_received[19] << 8) | data_received[18];
411.   Serial.print("Vibration 2: ");
412.   Serial.println(aux_u);
413.   Serial.print("Vibration 3: ");
414.   aux_u = 0;
415.   aux_u = (data_received[21] << 8) | data_received[20];
416.   Serial.println(aux_u);
417.   command_list_menu();
418. }
419.
420. void command_list_menu(void)
421. {
422.   Serial.println("-----------------------------------------");
423.   Serial.println("Command Line:                           |");
424.   Serial.println("Press T to Transmit Configuration       |");
425.   Serial.println("Press S to Set New Configuration        |");
426.   Serial.println("Press R to Read Present Configuration   |");
427.   Serial.println("Press D to Read Luminary Data            |");
428.   Serial.println("-----------------------------------------");
429. }
430.
431. void radio_config(void)
432. {
433.   radio.setChannel(0x60);
434.   radio.setRetries(15, 15);
435.   radio.setPALevel(RF24_PA_MAX);
```

```
436.   radio.setDataRate(RF24_250KBPS);
437.   radio.setPayloadSize(22);
438.   radio.setAutoAck(1);
439.   radio.setCRCLength(RF24_CRC_16);
440.   radio.openWritingPipe(pipes[1]);
441.   radio.openReadingPipe(1, pipes[0]);
442.   radio.printDetails();
443. }
444. 
445. void packet_creator(uint8_t id, uint8_t type)
446. {
447.   switch (type)
448.   {
449.     case CONFIGURATION_MSG:
450.       data2send[0] = my_node;
451.       data2send[1] = id;
452.       data2send[2] = 0; data2send[3] = 0; data2send[4] = 0; data2send[5] = 0;
453.       data2send[6] = type;
454.       data2send[7] = luminary_config.maxLight;
455.       data2send[8] = luminary_config.minLight;
456.       data2send[9] = (luminary_config.maxTime & 0x00FF);
457.       data2send[10] = (luminary_config.maxTime & 0xFF00) >> 8;
458.       data2send[11] = luminary_config.dimTime;
459.       data2send[12] = (luminary_config.minTime & 0x00FF);
460.       data2send[13] = (luminary_config.minTime & 0xFF00) >> 8;
461.       data2send[14] = luminary_config.PIR_gain;
462.       data2send[15] = luminary_config.PIR_thresh;
463.       data2send[16] = luminary_config.MW_thresh;
464.       data2send[17] = luminary_config.MW_Npulses;
465.       data2send[18] = luminary_config.ADXL_thresh;
466.       data2send[19] = luminary_config.infoTime;
467.       break;
468. 
469.     case READ_CONF_MSG:
470.       data2send[0] = my_node;
471.       data2send[1] = id;
472.       data2send[2] = 0; data2send[3] = 0; data2send[4] = 0; data2send[5] = 0;
473.       data2send[6] = type;
474.       break;
475. 
476.     case READ_DATA:
477.       data2send[0] = my_node;
478.       data2send[1] = id;
479.       data2send[2] = 0; data2send[3] = 0; data2send[4] = 0; data2send[5] = 0;
480.       data2send[6] = type;
481.       break;
482. 
483.     default:
484.       break;
485.   }
486. }
```